# EDC series Servo drive

# CANopen user's Manual

## Version: V1.01

## Estun Limited Warranty

# —— Contents ——

# 1、General introduction

## 1.1 CAN main related files

| Document Name | Source |
|---|---|
| **CiA DS 301 V 4.01:**<br>CANopen Communication Profile<br>for Industrial Systems - based on CAL | **CiA** |
| **CiA DSP 402 V 2.0:**<br>CANopen Device Profile | **CiA** |

## 1.2 Terms and Abbreviations Used in this Guide

**CAN** Controller Area Network

**CiA** CAN in Automation International Users and Manufacturers Group.

**COB** Communication Object; a unit of transportation on a CAN network. Data is sent across a network inside a COB. The COB itself is part of the CAN message frame.

**EDS** Electronic Data Sheet; a node-specific ASCII-format file required when configuring the CAN network. The EDS file contains general information on the node and its dictionary objects (parameters). EDS files for Estun drives are available through your local Estun sales agent.

**LMT** Layer Management; one of the service elements of the CAN Application Layer in the CAN Reference Model. It serves to configure parameters for each layer in the CAN Reference Model.

**NMT** Network Management; one of the service elements of the CAN Application Layer in the CAN Reference Model. It performs initialization, configuration and error handling on a CAN network.

**OD** Object Dictionary, to local-storage all communication objects identified by a certain equipment.

**Parameter** An operational instruction of driver, can be read and modified through CAN or driver digital operation panel.

**PDO** Process Data Object; a type of COB. Used for transmitting time-critical data, such as control commands, references and actual values.

**RO** Denotes read-only access.

**RW** Denotes read/write access.

**SDO** Service Data Object; a type of COB. Used for transmitting non-time critical data, such as parameters.

## 1.3 CANopen general introduction

CANopen is a higher-layer protocol based on the CAN (Control Area Network) serial bus system and the CAL (CAN Application Layer). CANopen assumes that the hardware of the connected device has a CAN transceiver and a CAN controller as specified in ISO 11898.

The CANopen Communication Profile, CiA DS-301, includes both cyclic and event-driven communication, which makes it possible to reduce the bus load to minimum while still maintaining extremely short reaction times. High communication performance can be achieved at relatively low baud rates, thus reducing EMC problems and cable costs.

CANopen device profiles define both direct access to drive parameter and time-critical process data communication. The NCAN-02 fulfils CiA (CAN in Automation) standard DSP-402 (Drives and Motion Control), supporting the 'Manufacturer Specific' operating mode only.

The physical medium of CANopen is a differentially-driven two-wire bus line with common return according to ISO 11898. The maximum length of the bus is limited by the communication speed as follows:

| Communication baud rate | Max. BUS length |
|---|---|
| 1M bit/s | 25 m |
| 500k bit/s | 100 m |
| 250k bit/s | 250 m |
| 125k bit/s | 500 m |
| 100k bit/s | 600 m |
| 50k bit/s | 1000 m |

The maximum theoretical number of nodes is 127. However, in practice, the maximum number depends on the capabilities of the CAN transceivers used.
Further information can be obtained from the CAN in Automation International Users and Manufacturers Group (*www.can-cia.de*).

# 2、wiring and connections

## ● EDC series appearance description


**Charge indicator**
Lights when the main circuit power supply is ON and stays lit as long as the main circuit power supply capacitor remains charged.

**Power ON indicator**
When the control power supply is ON,
The green light:the servo is in free run state;
The red light:the servo is in fault state.

**Rotary switch (ID)**

**Communication connector(CAN)**

**Communication connector(RS232)**
Used to communicate with a personal computer or to connect a digital operator.

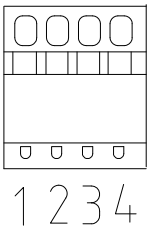
**Connector, 1CN for host connection**
Used for reference input signals and sequence I/O signals.

**Connector,2CN for encoder connection**
Connects to the encoder in the servomotor.

**Servomotor terminals**
Connects to the servomotor power line.

**Main power and Control power input terminals(single phase)**

## ● CAN communication plug interface and signal definistion

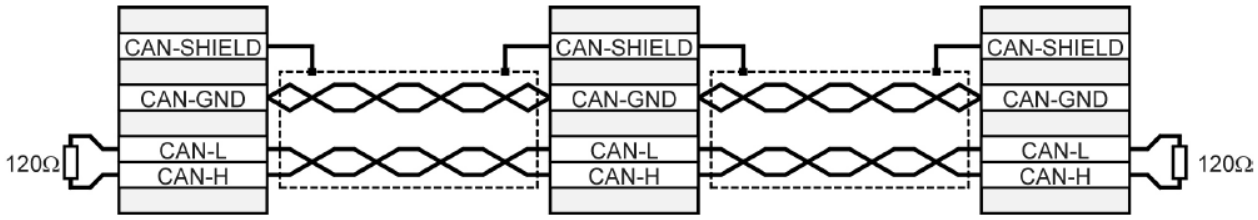



| Pin | definition |
|---|---|
| 1 | GND，internal grounding within servo drive |
| 2 | CANH |
| 3 | CANL |
| 4 | FG，shield grounding |

Rotary switch of the driver is used to set the communication address when communication through CAN or with PC.

When the rotary switch is 0, the RS232 port of driver could communicate with palm operator & through CAN.(Although it is 0, there is no such address in CAN. Therefore, CAN communication address is 1 under such kind of circumstance.) When the rotary switch is not 0, the RS232 port of driver could communicate with PC (Like using Esview software). Meanwhile, CAN communication is also available (The rotary switch address is the CAN communication address)

Note: When consist of CAN communication network, it is a must to connect a 120 Ohm resistor (1%,1/4W) as follows:




For cabling **shielded** cable with exactly two **twisted** pairs have to be used.

One twisted pair is used for CAN-H and CAN-L.

One twisted pair is used commonly for CAN-GND.

# 3、CANopen communication

CAL supplies all the network management service and message transport protocol. However, it didn't define the content of the object or the type of object that is communicating. ( It defines how instead of what), This is where CANopen could play an important role.

CANopen is based on CAL. Through CAL's communication and service protocol set, It supplies a solution to distributive control system. CANopen could ensure the interaction between network nodes and random extension of nodes' functions. It could be easy or complicated.

CANopen's core value is Object Dictionary. It is applied also in other field bus systems like Profibus and Interbus-S. CanOpen could access to all the parameters of the drive through object dictionary. Please notice that object dictionary is not one part of CAL., instead of which, it is realized in CANopen.

CANopen communication mode defines messages as below ( Communication objects) :

| Abbreviation | Full name | Description |
|---|---|---|
| **SDO** | Service Data Object | No real time but important data like parameters. |
| **PDO** | Process Data Object | Real time key process data( reference value, control word and status information) |
| **SYNC** | Synchronization Message | Synchronization of CAN node. |
| **EMCY** | Emergency Message | Alarm message transferring |
| **NMT** | Network Management | CANopen Network management |
| **Heartbeat** | Error Control Protocol | Supervising the availability of all nodes. |

CAN transmits data between host( controller) and the bus nodes through data frames. Struction of data frame is as below …

| Frame head | arbitration area | | control area | Data area | Checksum area | Response area | Frame tail |
|---|---|---|---|---|---|---|---|
| | COB-ID | RTR | | | | | |
| 1bit | 11or29bits | 1bit | 6bit | 0~8byte | 16 bits | 2 bits | 7 bits |

EDC doesn't support remote frames temporarily. COB-ID's structure is as below …

| Function code | | | | NODE ID（node address） | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

## 3.1 CAN identifier list

| Communication Object | Function code COB-ID bit10~7 (binary) | COB-ID (hex) | The reference of correspondent communication parameters in OD |
|---|---|---|---|
| NMT | 0000 | $000_h$ | — |
| SYNC | 0001 | $080_h$ | $1005_h$、$1006_h$、$1007_h$ |
| TIME STAMP | 0010 | $100_h$ | $1012_h$、$1013_h$ |
| EMCY | 0001 | $081_h$ ~ $0FF_h$ | $1024_h$、$1015_h$ |
| PDO1（send） | 0011 | $181_h$ ~ $1FF_h$ | $1800_h$ |
| PDO1（receive） | 0100 | $201_h$ ~ $27F_h$ | $1400_h$ |
| PDO2（send） | 0101 | $281_h$ ~ $2FF_h$ | $1801_h$ |
| PDO2（receive） | 0110 | $301_h$ ~ $37F_h$ | $1401_h$ |
| SDO（send） | 1011 | $581_h$ ~ $5FF_h$ | $1200_h$ |
| SDO（receive） | 1100 | $601_h$ ~ $67F_h$ | $1200_h$ |
| Heartbeat | 1110 | $701_h$ ~ $77F_h$ | $1016_h$、$1017_h$ |

Notice：

1、 PDO/SDO’s sending and receiving is detected by Can slave nodes.

## 3.2 SDO

SDO is used to visit the object dictionary of a device. A visitor is called a client. A CANopen device whose object dictionary is accessed will offer the required service and this devise is called a server. CAN messages from client or server always contain 8-bit data even if not each of them is meaningful). A client's requirement must have a answer from the server.

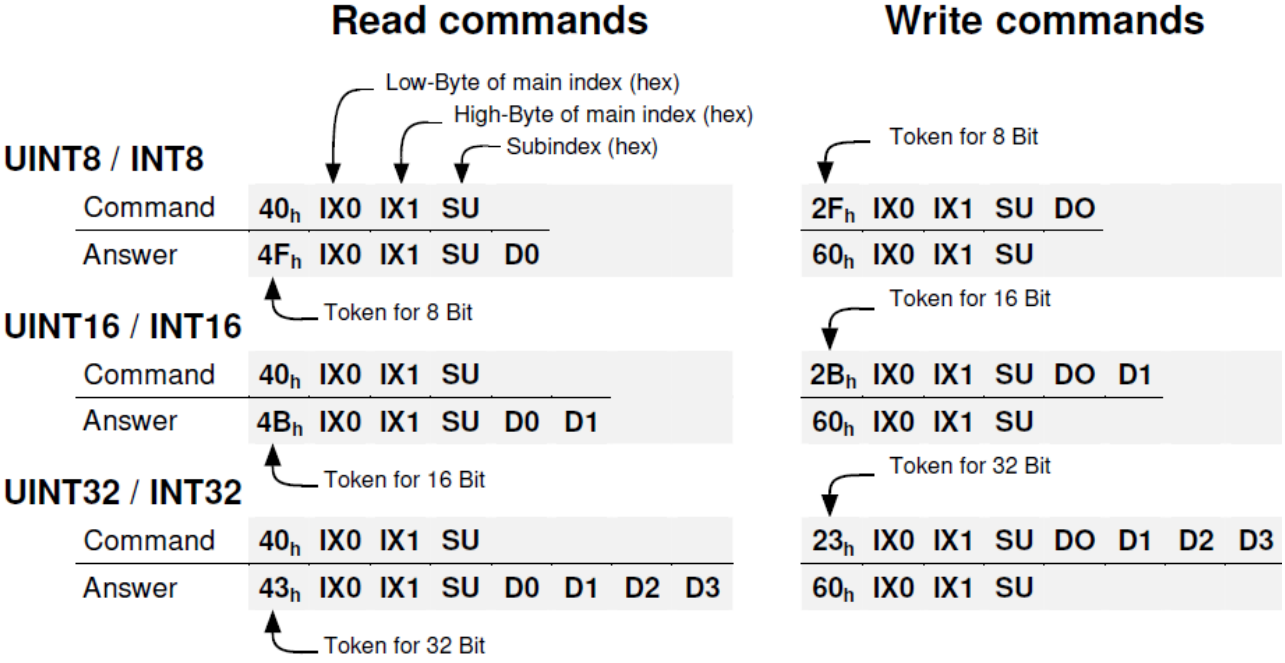
SDO has 2 transmitting mechanism.

- Expedited transfer ： 4 bytes at maximum to transfer
- Segmented transfer ： More than 4-byte data will be transmitted

SDO basic structure is as below …

| Byte0 | Byte1~2 | Byte3 | Byte4~7 |
|---|---|---|---|
| SDO command | Object reference | Object sub-reference | data |

SDO message's reading/writing frame …




For example:

**UINT8 / INT8**

Reading of Obj. $6061\_00_h$
Returning data: $01_h$

| | |
|---|---|
| Command: | $40_h$ $61_h$ $60_h$ $00_h$ |
| Answer: | $4F_h$ $61_h$ $60_h$ $00_h$ $01_h$ |

Writing of Obj. $1401\_02_h$
Data: $EF_h$

| | |
|---|---|
| Command: | $2F_h$ $01_h$ $14_h$ $02_h$ $EF_h$ |
| Answer: | $60_h$ $01_h$ $14_h$ $02_h$ |

**UINT16 / INT16**

Reading of Obj. $6041\_00_h$
Returning data: $1234_h$

| | |
|---|---|
| Command: | $40_h$ $41_h$ $60_h$ $00_h$ |
| Answer: | $4B_h$ $41_h$ $60_h$ $00_h$ $34_h$ $12_h$ |

Writing of Obj. $6040\_00_h$
Data: $03E8_h$

| | |
|---|---|
| Command: | $2B_h$ $40_h$ $60_h$ $00_h$ $E8_h$ $03_h$ |
| Answer: | $60_h$ $40_h$ $60_h$ $00_h$ |

**UINT32 / INT32**

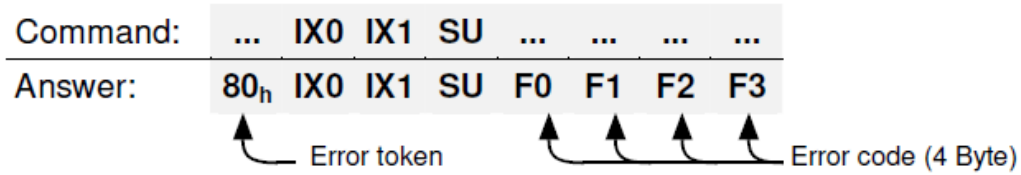
Reading of Obj. $6093\_01_h$
Returning data: $12345678_h$

| | |
|---|---|
| Command: | $40_h$ $93_h$ $60_h$ $01_h$ |
| Answer: | $43_h$ $93_h$ $60_h$ $01_h$ $78_h$ $56_h$ $34_h$ $12_h$ |

Writing of Obj. $6093\_01_h$
Data: $12345678_h$

| | |
|---|---|
| Command: | $23_h$ $93_h$ $60_h$ $01_h$ $78_h$ $56_h$ $34_h$ $12_h$ |
| Answer: | $60_h$ $93_h$ $60_h$ $01_h$ |

SDO error message frame




| Error code F3 F2 F1 F0 | Description |
|---|---|
| 05 03 00 00$_h$ | Toggle bit not alternated |
| 05 04 00 01$_h$ | Client / server command specifier not valid or unknown |
| 06 01 00 00$_h$ | Unsupported access to an object |
| 06 01 00 01$_h$ | Attempt to read a write only object |
| 06 01 00 02$_h$ | Attempt to write a read only object |
| 06 02 00 00$_h$ | Object does not exist in the object dictionary |
| 06 04 00 41$_h$ | Object cannot be mapped to the PDO |
| 06 04 00 42$_h$ | The number and length of the objects to be mapped would exceed PDO length |
| 06 04 00 47$_h$ | General internal incompatibility in the device |
| 06 07 00 10$_h$ | Data type does not match, length of service parameter does not match |
| 06 07 00 12$_h$ | Data type does not match, length of service parameter too high |
| 06 07 00 13$_h$ | Data type does not match, length of service parameter too low |
| 06 09 00 11$_h$ | Sub-index does not exist |
| 06 04 00 43$_h$ | General parameter incompatibility |
| 06 06 00 00$_h$ | Access failed due to an hardware error *1) |
| 06 09 00 30$_h$ | Value range of parameter exceeded |
| 06 09 00 31$_h$ | Value of parameter written too high |
| 06 09 00 32$_h$ | Value of parameter written too low |
| 06 09 00 36$_h$ | Maximum value is less than minimum value |
| 08 00 00 20$_h$ | Data cannot be transferred or stored to the application *1) |
| 08 00 00 21$_h$ | Data cannot be transferred or stored to the application because of local control |
| 08 00 00 22$_h$ | Data cannot be transferred or stored to the application because of the present device state *3) |
| 08 00 00 23$_h$ | No Object Dictionary is present *2) |

## 3.3 PDO

PDO is used to transmit real-time data which is from a data creator to multiple data consumers. Transmitting data is limited from 1 byte to 8 bytes. PDO communication is not limited by any protocols, which means the content of data has already been pre-defined. As a result, consumers could finish processing received data in a very short period. PDO data is only defined by its CAN ID, assuming both data creators and data consumers know the content of PDO.

Every PDO is described by 2 objects in object dictionary.

 PDO communication parameter： It contains COB-ID, transmitting type, frozen time, period of timer which are all used by PDO.

 PDO mapping parameters : It contains the object list in one object dictionary. All this objects are mapped to PDO, including their data length (in bits). Data creators and consumers must know this mapping to describe the content of PDO.

The content of PDO is pre-defined or pre-configured when the network is initialized. Mapping the application objects to the PDO is described in object dictionary. If device( data creators and consumers) supports dynamism, SDO messages could be used to configure the PDO mapping parameters. EDC could support PDO mapping. 2 rules of PDO mapping have to be obeyed as below …

1、 One PDO could be used to map 4 objects at maximum.

2、 The length of each PDO has to be 64 bits or below.

PDO mapping procedures.

1、 Setting the correspondent mapping parameters of PDO. （$1600_h$、$1601_h$、$1602_h$、$1603_h$ 或 $1A00_h$、$1A01_h$、$1A02_h$、$1A03_h$_. The content of sub-reference, 0, is o.

2、 Revise the content of sub-index 1~4 （$1600_h$、$1601_h$、$1602_h$、$1603_h$ and $1A00_h$、$1A01_h$、$1A02_h$、$1A03_h$）which are PDO's correspondent mapping parameters.

3、 Set the content of sub-index 0 of PDO correspondent mapping parameters as legal figures. ( number of PDO's mapping objects)

4、 PDO mapping completed

PDO could be transmitted in multiple ways ：

 Synchronous transmitting（Synchronization through accepting SYNC objects）

Period: Transmitting will be triggered after 1 to 240 SYNC messages.

 Asynchronous transmitting

Special object incident defined in device sub-protocol could trigger the transmitting

PDO transmit defining list

| **Transmit type value** | **Description** | **PDO** |
|---|---|---|
| 0 | reserved | — |
| 1~240 | **SYNC method：the number of SYNC objects between 2 PDOs** | TPDO/RPDO |
| 240~253 | reserved | — |
| 254 | **Asynchronous method：If the content of PDO changes, it will trigger PDO.** | TPDO |
| 255 | **Asynchronous method:** cyclical update and sending of PDO content | TPDO/RPDO |

One PDO could settle a frozen time, that is, the minimum time between 2 continuous PDOs, which could avoid the high preferential information with big data volume keeps occupying the bus and other information with low priority will be unable to compete for bus resource. Frozen time is settled by 16-bit unsigned integrals whose unit is 100us.

One PDO could settle an incident timing period. When passing the regulated time, one PDO sending could be triggered without a trigger bit. Incident timing period could be defined by 16-bit unsigned integral whose unit is 1ms.

PDO mapping：

Map the three objects as below to PDO1(sending). PDO1( sending) is asynchronous cyclical type. The cycle time is 10ms and the frozen time is 2ms.

| objects | reference — sub-reference | instruction |
|---|---|---|
| statusword | $6041_{h}$ – 00 $_{h}$ | Status word |
| modes_of_operation_display | $6061_{h}$ – 00 $_{h}$ | actual operation mode |
| Position_Acture_Value | $6064_{h}$ – 00 $_{h}$ | Actual position |

1）、clear number_of_mapped_objects

number_of_mapped_objects(10A0 $_{h}$： 00 $_{h}$)= 0

2）、setting mapping object parameter

Index =6041 $_{h}$ Subin. = 00h Length = 10 $_{h}$ □ 1st_mapped_object(10A0 $_{h}$： 01 $_{h}$)= 60410010 $_{h}$

Index =6061 $_{h}$ Subin. = 00h Length = 08 $_{h}$ □ 2st_mapped_object(10A0 $_{h}$： 02 $_{h}$)= 60610008 $_{h}$

Index =60FD $_{h}$ Subin. = 00h Length = 20 $_{h}$ □ 3st_mapped_object(10A0 $_{h}$： 03 $_{h}$) = 60FD0020 $_{h}$

3）、setting number_of_mapped_objects

number_of_mapped_objects(10A0 $_{h}$： 00 $_{h}$)= 3

4）、setting PDO communication parameter

PDO1（sending）is asynchronous type □ transmission_type (1800 $_{h}$： 02 $_{h}$)= FF $_{h}$

Frozen time 2ms(20×100us) □ inhibit_time (10A0 $_{h}$： 03 $_{h}$)= 14 $_{h}$

Cycle time10ms(10×1ms) □ event_time (1800 $_{h}$： 05 $_{h}$)= 0A $_{h}$

5）、PDO mapping is completed.

## 3.3.1 PDO parameter

EDC servo drive contains 4 sending PDOs and 4 receiving PDOs. The specification of communication parameters and mapping parameters for the first sending/receiving PDO is as below. The other 3 sending/receving PDO specifications are the same as the first one.

| | |
|---|---|
| Index | 1800 **h** |
| Name | transmit_pdo_parameter_tpdo1 |
| Object Code | RECORD |
| No. of Elements | 4 |

| | |
|---|---|
| Sub-Index | 01 **h** |
| Description | cob_id_used_by_pdo_tpdo1 |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | 181 **h**...1FF **h**, Bit 31 may be set |
| Default Value | 181 **h** |

| | |
|---|---|
| Sub-Index | 02 **h** |
| Description | transmission_type_tpdo1 |
| Data Type | UINT8 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | 1...240,254,255 |
| Default Value | 255 |

| | |
|---|---|
| Sub-Index | 03 **h** |
| Description | inhibit_time_tpdo1 |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | 100μs |
| Value Range | —— |
| Default Value | 100 |

| Sub-Index | 05 h |
|---|---|
| Description | event_time_tpdo1 |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | 1ms |
| Value Range | —— |
| Default Value | 10 |

| Index | 1A00 h |
|---|---|
| Name | transmit_pdo_mapping_tpdo1 |
| Object Code | RECORD |
| No. of Elements | 2 |

| Sub-Index | 00 h |
|---|---|
| Description | number_of_mapped_objects_tpdo1 |
| Data Type | UINT8 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | 0...4 |
| Default Value | 2 |

| Sub-Index | 01 h |
|---|---|
| Description | first_mapped_object_tpdo1 |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Please refer the list below |

| Sub-Index | 02 h |
|---|---|
| Description | second_mapped_object_tpdo1 |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Please refer the list below |

| Sub-Index | 03 **h** |
|---|---|
| Description | third_mapped_object_tpdo1 |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Please refer the list below |

| Sub-Index | 04 **h** |
|---|---|
| Description | fourth_mapped_object_tpdo1 |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Please refer the list below |

**RPDO（EDC software version>3.10）**

1、R-PDO1

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1400 h _00 h | number of entries | UINT8 | RO | 02 h |
| 1400 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000201 h |
| 1400 h _02 h | transmission type | UINT8 | RW | FF h |
| 1600 h _00 h | number of mapped objects | UINT8 | RW | 01 h |
| 1600 h _01 h | first mapped object | UINT32 | RW | 60400010 h |
| 1600 h _02 h | second mapped object | UINT32 | RW | 60FF0020 h |
| 1600 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1600 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

2、R-PDO2

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1401 h _00 h | number of entries | UINT8 | RO | 02 h |
| 1401 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000301 h |
| 1401 h _02 h | transmission type | UINT8 | RW | FF h |
| 1601 h _00 h | number of mapped objects | UINT8 | RW | 02 h |
| 1601 h _01 h | first mapped object | UINT32 | RW | 607A0020 h |
| 1601 h _02 h | second mapped object | UINT32 | RW | 60810020 h |
| 1601 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1601 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

3、R-PDO3

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1402 h _00 h | number of entries | UINT8 | RO | 02 h |
| 1402 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000401 h |
| 1402 h _02 h | transmission type | UINT8 | RW | FF h |
| 1602 h _00 h | number of mapped objects | UINT8 | RW | 00 h |
| 1602 h _01 h | first mapped object | UINT32 | RW | 60400010 h |
| 1602 h _02 h | second mapped object | UINT32 | RW | 00000000 h |
| 1602 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1602 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

4、R-PDO4

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1403 h _00 h | number of entries | UINT8 | RO | 02 h |
| 1403 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000501 h |
| 1403 h _02 h | transmission type | UINT8 | RW | FF h |
| 1603 h _00 h | number of mapped objects | UINT8 | RW | 00 h |
| 1603 h _01 h | first mapped object | UINT32 | RW | 607A0020 h |
| 1603 h _02 h | second mapped object | UINT32 | RW | 60810020 h |
| 1603 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1603 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

**TPDO**

1、T-PDO1

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1800 h _00 h | number of entries | UINT8 | RO | 04 h |
| 1800 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000181 h |
| 1800 h _02 h | transmission type | UINT8 | RW | FF h |
| 1800 h _03 h | inhibit time (100 μs) | UINT16 | RW | 64 h |
| 1800 h _05 h | event time (1ms) | UINT16 | RW | 0A h |
| 1A00 h _00 h | number of mapped objects | UINT8 | RW | 01 h |
| 1A00 h _01 h | first mapped object | UINT32 | RW | 60410010 h |
| 1A00 h _02 h | second mapped object | UINT32 | RW | 60640020 h |
| 1A00 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1A00 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

2、T-PDO2

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1801 h _00 h | number of entries | UINT8 | RO | 04 h |
| 1801 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000281 h |
| 1801 h _02 h | transmission type | UINT8 | RW | FF h |
| 1801 h _03 h | inhibit time (100 μs) | UINT16 | RW | 64 h |
| 1801 h _05 h | event time (1ms) | UINT16 | RW | 0A h |
| 1A01 h _00 h | number of mapped objects | UINT8 | RW | 02 h |
| 1A01 h _01 h | first mapped object | UINT32 | RW | 60640020 h |
| 1A01 h _02 h | second mapped object | UINT32 | RW | 606C0020 h |
| 1A01 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1A01 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

3、T-PDO3

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1802 h _00 h | number of entries | UINT8 | RO | 04 h |
| 1802 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000381 h |
| 1802 h _02 h | transmission type | UINT8 | RW | FF h |
| 1802 h _03 h | inhibit time (100 μs) | UINT16 | RW | 64 h |
| 1802 h _05 h | event time (1ms) | UINT16 | RW | 0A h |
| 1A02 h _00 h | number of mapped objects | UINT8 | RW | 00 h |
| 1A02 h _01 h | first mapped object | UINT32 | RW | 60410010 h |
| 1A02 h _02 h | second mapped object | UINT32 | RW | 00000000 h |
| 1A02 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1A02 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

4、T-PDO4

| Index | Comment | Type | Acc. | Default Value |
|---|---|---|---|---|
| 1803 h _00 h | number of entries | UINT8 | RO | 04 h |
| 1803 h _01 h | COB-ID used by PDO | UINT32 | RW | 00000481 h |
| 1803 h _02 h | transmission type | UINT8 | RW | FF h |
| 1803 h _03 h | inhibit time (100 μs) | UINT16 | RW | 64 h |
| 1803 h _05 h | event time (1ms) | UINT16 | RW | 0A h |
| 1A03 h _00 h | number of mapped objects | UINT8 | RW | 00 h |
| 1A03 h _01 h | first mapped object | UINT32 | RW | 60640020 h |
| 1A03 h _02 h | second mapped object | UINT32 | RW | 606C0020 h |
| 1A03 h _03 h | third mapped object | UINT32 | RW | 00 h |
| 1A03 h _04 h | fourth mapped object | UINT32 | RW | 00 h |

## 3.4 SYNC message

Synchronization in the network. Any input into the network will be preserved, and then transmitted if necessary. Output will be updated according to the last SYNC message.

Host-slave mode: SYNC host node will send SYNC objects during each certain period. SYNC slave node will execute SYNC mission after receiving the message.

CANopne advises to use a COB-ID with the most advanced priority to ensure the proper transmitting of synchronized signal. SYNC message could choose not to transmit data to shorten the message.

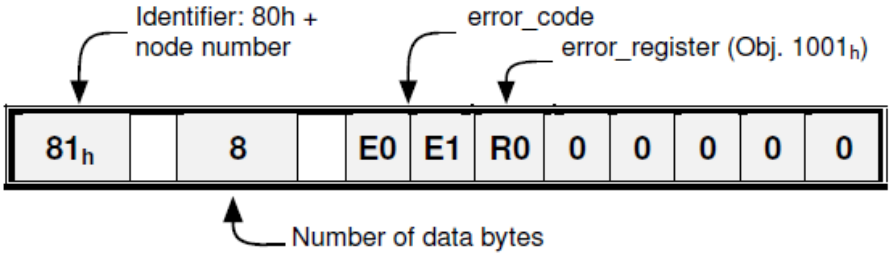
COB-ID of SYNC message is fixed to be 080h. COB-ID could be read from 1005 h in object dictionary.

| | |
|---|---|
| Index | 1005 h |
| Name | cob_id_sync |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | 80000080 h, 00000080 h |
| Default Value | 00000080 h |

## 3.5 Emergency message

When one alarm happens, CANopen will activate an Emergency message to inform the consumers about the current drive type and error code.

Emergency message structure：




Alarm code

| error_code（hex） | instruction |
|---|---|
| 2310 | Over current |
| 3100 | Sudden power loss |
| 3110 | extraordinary voltage |
| 3130 | No power input |
| 5080 | RAM chip abnormality |
| 5210 | AD sampling error |
| 5581 | Parameter checksum error |
| 5583 | parameter of motor or drive’s type error |
| 6100 | drive program error |
| 6300 | CAN communication parameter effort( address or communication baud rate error) |
| 7305 | incremental encoder error |
| 8081 | negative direction movement limited |
| 8082 | Positive direction movement limited |
| 8100 | CAN communication error |
| 8110 | CAN communication error |
| 8120 | CAN communication error |
| 8181 | CAN communication error |
| 8182 | CAN communication error |
| 8130 | Heartbeat error |
| 8200 | Length of CAN receiving message error |
| 8210 | Length of receiving PDO error |
| 8311 | Overload alarm |
| 8480 | Over speed alarm |

Details of parameters

| | |
|---|---|
| Index | 1003 **h** |
| Name | pre_defined_error_field |
| Object Code | ARRAY |
| No. of Elements | 4 |
| Data Type | UINT32 |

| | |
|---|---|
| Sub-Index | 01 **h** |
| Description | standard_error_field_0 |
| Access | RO |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | —— |

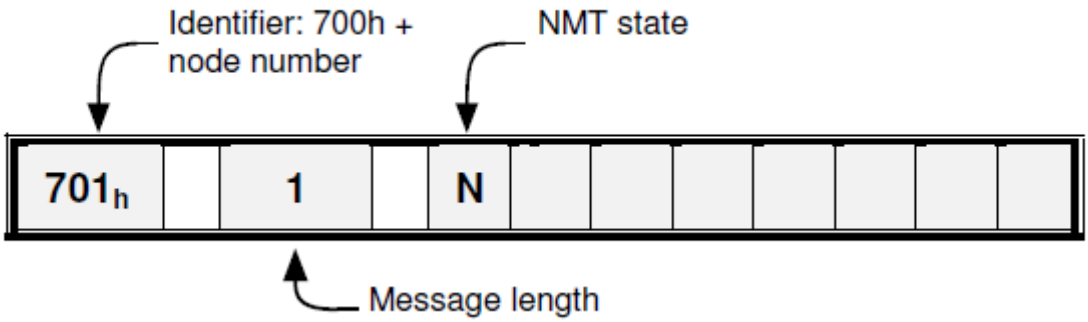
| | |
|---|---|
| Sub-Index | 02 **h** |
| Description | standard_error_field_1 |
| Access | RO |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | —— |

| | |
|---|---|
| Sub-Index | 03 **h** |
| Description | standard_error_field_2 |
| Access | RO |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | —— |

| | |
|---|---|
| Sub-Index | 04 **h** |
| Description | standard_error_field_3 |
| Access | RO |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | —— |

## 3.6 Heartbeat message

Message structure：

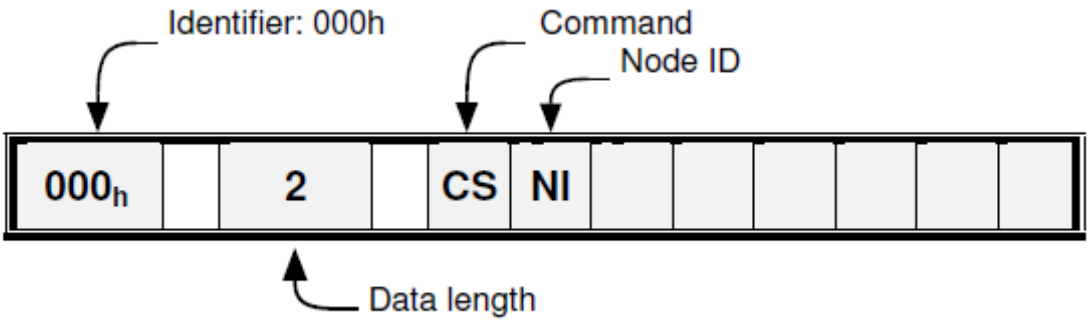



Details：

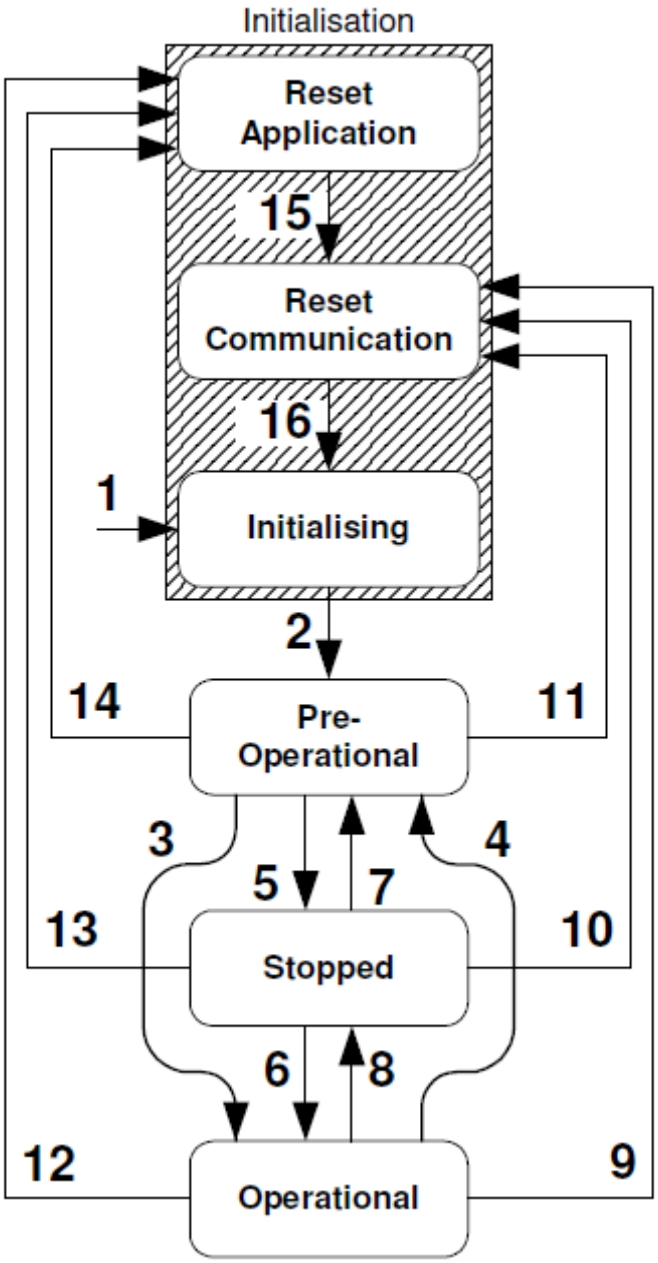
| | |
|---|---|
| Index | 1017 h |
| Name | producer_heartbeat_time |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | ms |
| Value Range | 0 - 65535 |
| Default Value | 0 |

## 3.7 Network management（NMT）

Message structure：




Network management status conversion graph:

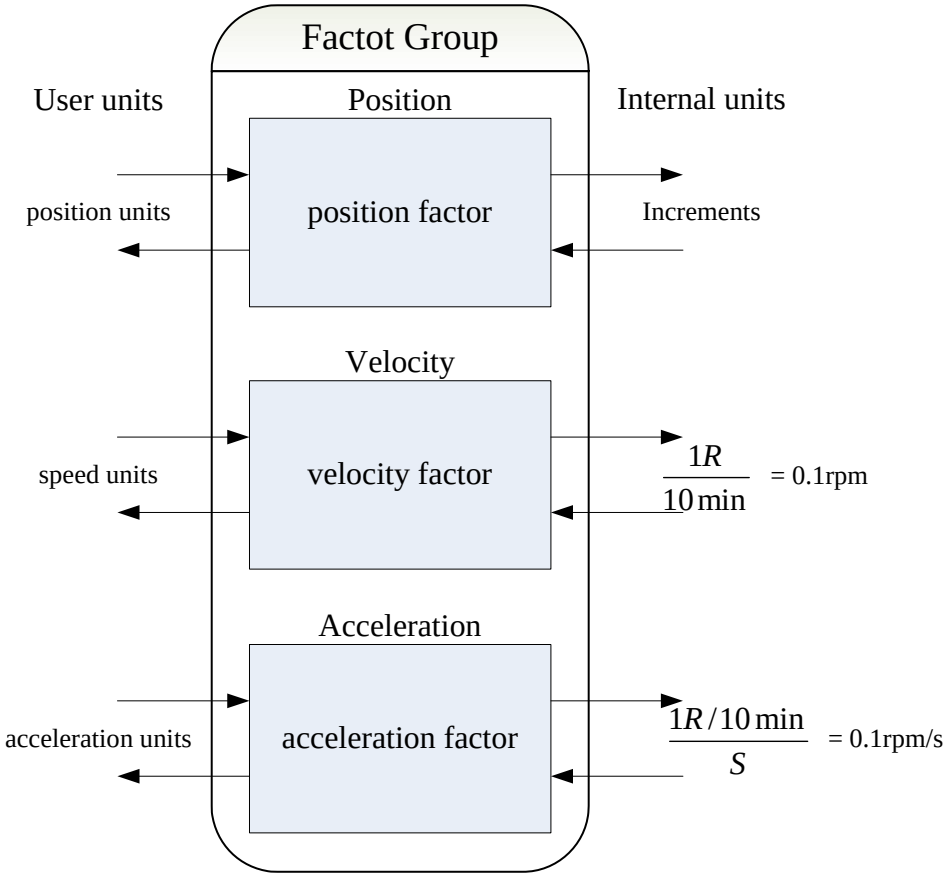



| CS | Meaning | Transition | Target state |
|---|---|---|---|
| $01_h$ | Start Remote Node | 3, 6 | **Operational** |
| $02_h$ | Stop Remote Node | 5, 8 | **Stopped** |
| $80_h$ | Enter Pre-Operational | 4, 7 | **Pre-Operational** |
| $81_h$ | Reset Application | 12, 13, 14 | **Reset Application** |
| $82_h$ | Reset Communication | 9, 10, 11 | **Reset Communication** |

| Name | Meaning | SDO | PDO | NMT |
|---|---|---|---|---|
| **Reset Application** | No communication. All CAN objects are set to their reset values (application parameter set). | - | - | - |
| **Reset Communication** | No communication.<br>The CAN controller will be re-initialised. | - | - | - |
| **Initialising** | State after Hardware Reset. Reset of the CAN node, sending of the Bootup message | - | - | - |
| **Pre-Operational** | Communication via SDOs possible.<br>PDOs inactive (No sending / receiving) | X | - | X |
| **Operational** | Communication via SDOs possible.<br>PDOs active (sending / receiving) | X | X | X |
| **Stopped** | No communication except heartbeat + NMT | - | - | X |

# 4、measuring unit conversion（Factor Group）

Servo drives are widely used in different applications. For setting parameters easily in different applications, our clients could use the internal measuring unit conversion module to converse any users' parameters into drive's internal unit.




：Default internal unit.:

| Object | name | unit | instruction |
|---|---|---|---|
| length | position units | Increments | 脉冲 * |
| speed | speed units | 1R /10min | 0.1rpm |
| acceleration | Acceleration units | 1R/10min/s | 0.1rpm/s |

**Note:**Normal incremental encoder will output 10000 pulses every revolution.

## 4.1 Measuring unit conversion parameter:

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 6093 h | ARRAY | position factor | UINT32 | RW |
| 6094 h | ARRAY | velocity factor | UINT32 | RW |
| 6097 h | ARRAY | acceleration factor | UINT32 | RW |

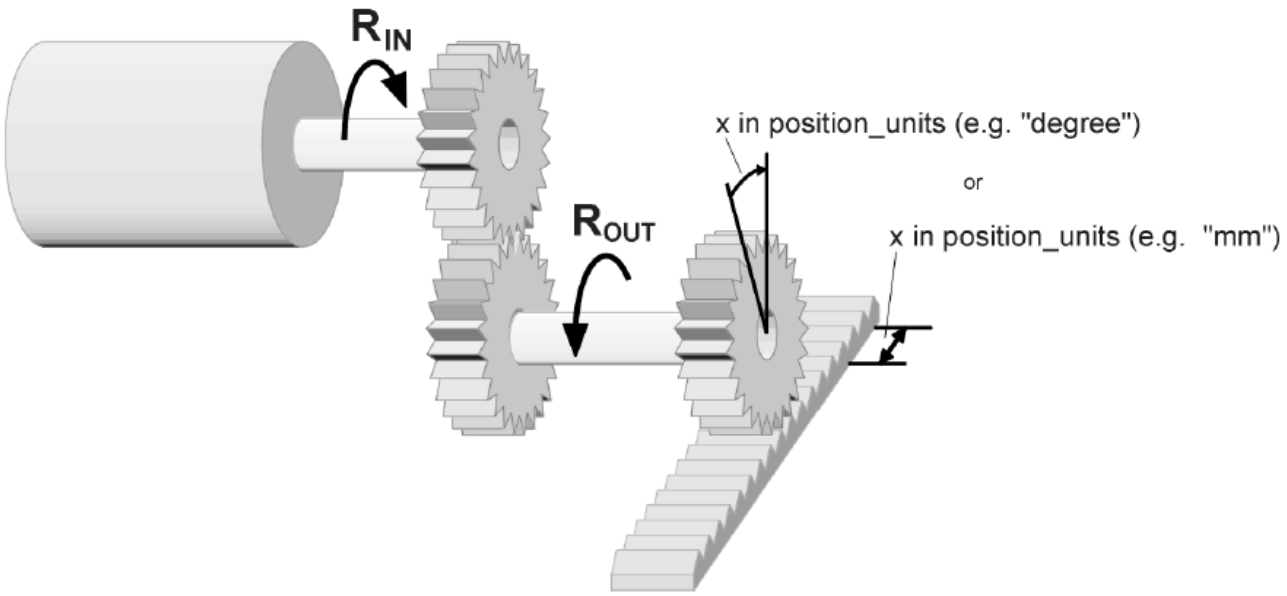
## 4.1.1 Position factor

Position factor module could convert all the measuring units of client into internal unit of servo drive (pulse) and at the same time convert the unit (pulse) of all the output from the drive into the measuring unit of clients (position units) Position factors includes numerator and division.

| | |
|---|---|
| Index | 6093 h |
| Name | position factor |
| Object Code | ARRAY |
| No. of Elements | 2 |
| Data Type | UINT32 |

| | |
|---|---|
| Sub-Index | 01 h |
| Description | numerator |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Initialized to 1 when power on |

| | |
|---|---|
| Sub-Index | 02 h |
| Description | division |
| Access | RW |
| PDO Mapping | NO |
| Units | —— |
| Value Range | —— |
| Default Value | Initialized to 1 when power on |




For calculating the position factors easily, 2 parameters as below are defined …

**gear_ratio** Reduction ration between the load shaft and the motor shaft.
（when motor's revolution is n and load's revolution is m，then **gear_ratio** = m/n）

**feed_constant** the distance of position units' movement when load shaft rotates for one revolution.

position factor's calculating equation …

$$\textbf{position factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{encoder\_resolution}}{\text{feed\_constant}}$$

Note：

| Encoder type | encoder_resolution(Unit: Inc) |
|---|---|
| Normal incremental encoder | 10000 |

## 4.1.2 Velocity factor

Velocity factor module will convert all the speed measuring unit at customer side into drive's internal measuring unit as much as 0.1rpm. And at the same time, it could transform the drive's output velocity unit (0.1rpm) into user's velocity units. Velocity factor parameters includes a numerator and a division.

| Index | 6094 **h** |
|---|---|
| Name | velocity factor |
| Object Code | ARRAY |
| No. of Elements | 2 |
| Data Type | UINT32 |

| Sub-Index | 01 **h** |
|---|---|
| Description | numerator |
| Access | RW |
| PDO Mapping | YES |
| Units | —— |
| Value Range | —— |
| Default Value | 1 |

| Sub-Index | 02 **h** |
|---|---|
| Description | division |
| Access | RW |
| PDO Mapping | YES |
| Units | —— |
| Value Range | —— |
| Default Value | 1 |

For calculating velocity factor easily, 3 parameters are defined as below …

**time_factor_v** Ratio between drive's internal time unit and user's time unit. ( For example: 1**min** = 1/10 **10min**）

**gear_ratio** reduction ratio between load shaft and motor shaft

when motor's revolution is n and load's revolution is m，then **gear_ratio** = m/n

**feed_constant** the distance of position units' movement when load shaft rotates for one revolution.

velocity factor's calculating equation …

$$\textbf{velocity factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{time\_factor\_v}}{\text{feed\_constant}}$$

## 4.1.3 Acceleration factor

Acceleration factor module will convert all the acceleration units at the perspective of clients into drive's internal unit (0.1rpm) and at the same time, converts output acceleration units (0.1rpm) from the drive into acceleration units at the perspective of clients. Acceleration factor parameters contain numerator and division.

| Index | 6094 **h** |
|---|---|
| Name | acceleration factor |
| Object Code | ARRAY |
| No. of Elements | 2 |
| Data Type | UINT32 |

| Sub-Index | 01 **h** |
|---|---|
| Description | numerator |
| Access | RW |
| PDO Mapping | YES |
| Units | —— |
| Value Range | —— |
| Default Value | 1 |

| Sub-Index | 02 **h** |
|---|---|
| Description | division |
| Access | RW |
| PDO Mapping | YES |
| Units | —— |
| Value Range | —— |
| Default Value | 1 |

For calculating velocity factor easily, we could define 3 variables as below …

**time_factor_a** The ratio between drive's internal time square and clients' time square.
（For example： 1**min²** = 1min*min = 60s*1min =60/10 **10min/s**）

**gear_ratio** reduction ratio between load shaft and motor shaft
when motor's revolution is n and load's revolution is m，then **gear_ratio** = m/n

**feed_constant** the distance of position units' movement when load shaft rotates for one revolution.

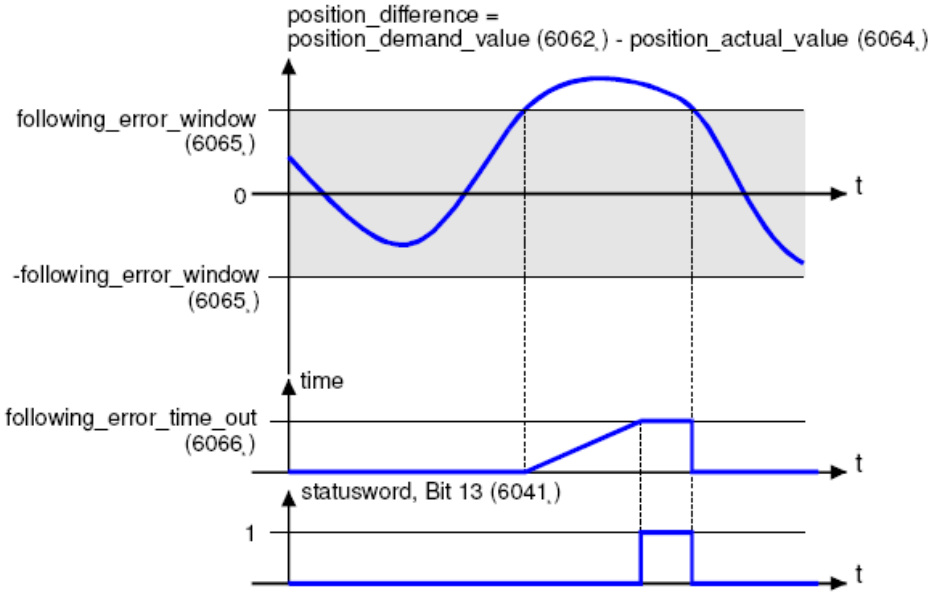
$$\textbf{acceleration factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{time\_factor\_a}}{\text{feed\_constant}}$$

# 5、Position control function

This chapter mainly describes the parameters under position control mode. Trajectory unit will output reference position (position_demand_value) will be the input of drive's position loop. Besides, the actual position( position_actual_value) is measured through the motor's encoder. Position control will be influenced by parameter setting. For Stabilizing the control system, we have to limit the output of postion loop ( control_effect). This output will become the fixed speed for speed loop. In Factor group, all the input and output will be transformed into the internal measuring unit of the servo drive.

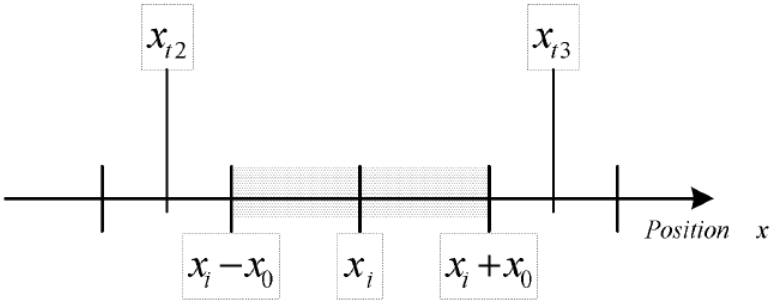
Below is the introduction of sub-function for position control.

1、following error




following error –function description

Following error is the deviation between actual position (position_actual_value) and reference position (position_demand_value). As above, if within following_error_time_out the following deviation is bigger than following error windows, the bit 13 of status word, that is, following_error will be set as 1.



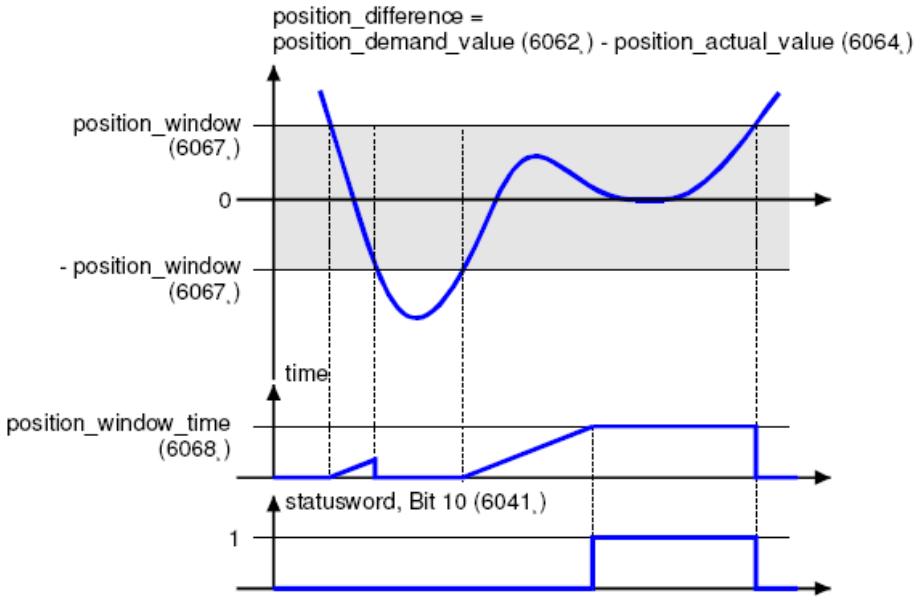

following error–for example

Above is the description about how to define window function as to following error. xi-x0and xi+x0 (following error window)are located symmetrically at each side of position_demand_value. For example,

xt2 and xt3 are both out of following error window. If the drive leaves the window and doesn't return back to the window in following_error_time_out, bit 13(following_error) of the statusword will be set as much as 1.

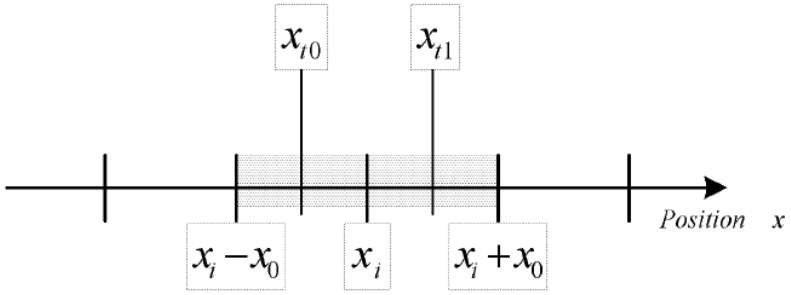
2、Position reached.

This function defines the position window near target position. If the servo drive's actual position is set stably in the position window, bit 10 of status word (target_reached) will be set as 1.




Position reached-function description

As below, position_windows is located symmetrically at places near target_position, that is, between xi-x0 and xi+x0. For example, xt0 and xt1 are in the position_windows. If the servo drive is in the windows, one timer starts working. If the timer reach position_window_time when the servo drive is in the windows, bit10 of statusword , target_reached, will bet set as much as 1. Once the servo drive leaves this window, bit10(target_reached) of the status will be cleared to 0.




Position reached-example

# 5.1 Parameters about position control

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 6062 h | VAR | position_demand_value | INT32 | RO |
| 6063 h | VAR | position_actual_value* | INT32 | RO |
| 6064 h | VAR | position_actual_value | INT32 | RO |
| 6065 h | VAR | following_error_window | UINT32 | RW |
| 6066 h | VAR | following_error_time_out | UINT16 | RW |
| 6067 h | VAR | position_window | UINT32 | RW |
| 6068 h | VAR | position_time | UINT16 | RW |

| | |
|---|---|
| Index | 6062 h |
| Name | position_demand_value |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RO |
| PDO Mapping | YES |
| Units | position units |
| Value Range | -- |
| Default Value | -- |

| | |
|---|---|
| Index | 6064 h |
| Name | position_ actual _value |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RO |
| PDO Mapping | YES |
| Units | position units |
| Value Range | -- |
| Default Value | -- |

| | |
|---|---|
| Index | 6065 h |
| Name | following_error_window |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | position units |
| Value Range | 0 – 7FFFFFFF h |
| Default Value | 30000 |

| Index | 6066 h |
|---|---|
| Name | following_error_time_out |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | ms |
| Value Range | 0 – 65535 |
| Default Value | 200 |

| Index | 6067 h |
|---|---|
| Name | position_window |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | NO |
| Units | position units |
| Value Range | -- |
| Default Value | 10 |

| Index | 6068 h |
|---|---|
| Name | position_time |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | ms |
| Value Range | 0 – 65535 |
| Default Value | 100 |

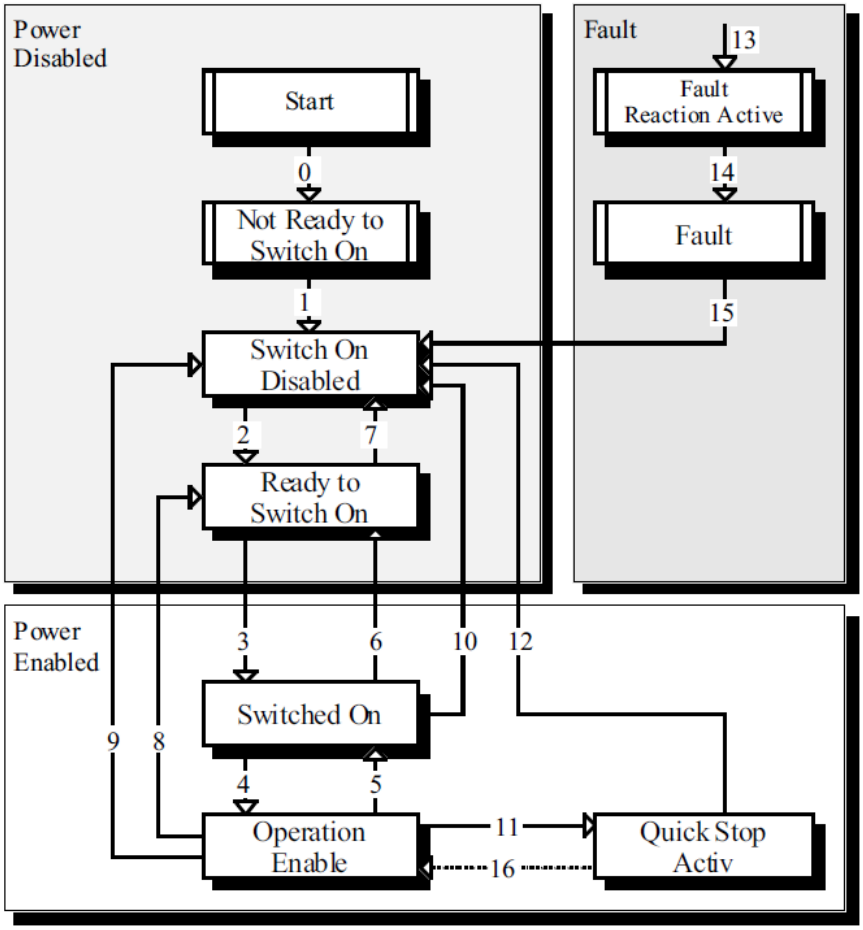
# 6、Device control

This chapter describes how the host could control the servo drive through CANopen, like enabling servo on and clearing alarms.

## 6.1 Control state machine.

The host could control the servo drive through controlword. It could know the current status of the servo drive through reading the statusword of the servo drive. This chapters will use the terms as below ...

| | |
|---|---|
| State： | When the main circuit is activated or alarms happen, the servo drive is in different status. This chapter is mainly about the state machine controlled by CANopen<br>For example：SWITCH_ON_DISABLED |
| State Transition： | Status machine also describes how to transit from one state to another state. State transition mainly relies on controlword controlled by the host or servo drive itself, for example, alarm. |
| Command： | For initialling State Transition, the bit composition of control word is defined. This bit composition is called Command. |
| State diagram： | All the States and State Transitions compose a State diagram. |




State machine graph

As above, state machine could be divided into 3 parts, "Power Disabled", "Power Enabled" and "Faullt". All the states will transit to "Fault" after any alarm happens.After power on, the servo drive will finish initializing and enter the state of SWITCH_ON_DISABLED. Under this state, CAN communication and servo drive configuration( for example, setting the work mode of the servo drive as PP mode) are still available. At then, the main circuit is still shut down and motor is out of excitation. After state Transition 2, 3 and 4, it become OPERATION ENABLE. And then, the main circuit has been initialized and servo drive will control the servo motor according to the configured work mode. Hence, we have to confirm we have configured the servo drive's parameters correctly and we have set correspondent input value as 0 before this state. State Transition 9 will shut down the power supply to the main circuit. Once any alarms happens to the servo drive, the state of the servo drive will enter FAULT. state

| State | Instruction |
|---|---|
| **Not Ready to Switch On** | The servo drive is on the way of initialization and no CAN communication is available. |
| **Switch On Disabled** | Initialization is completed and CAN communication is available now. |
| **Ready to Switch On** | Servo drive waits to enter Switch On state and the servo motor is out of excitation. |
| **Switched On** | |
| **Operation Enable** | The servo drive is inputing excitation signal to the servo motor and control the servo motor according to the control mode. |
| **Quick Stop Active** | Servo drive will stop according to the set method. |
| **Fault Reaction Active** | Alarm detects and servo motor is out excitation. |
| **Fault** | |

## 6.2 Parameters for device control

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 6040 h | VAR | controlword | UINT16 | RW |
| 6041 h | VAR | statusword | UINT16 | RO |
| 605A h | VAR | quick_stop_option_code | INT16 | RW |
| 605B h | VAR | shutdown_option_code | INT16 | RW |
| 605C h | VAR | disabled_operation_option_code | INT16 | RW |
| 605D h | VAR | halt_option_code | INT16 | RW |
| 605E h | VAR | fault_reaction_option_code | INT16 | RW |

## 6.2.1 Control word

| | |
|---|---|
| Index | 6040 h |
| Name | controlword |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | -- |
| Value Range | -- |
| Default Value | 0 |

Controlword bit explanation is as below ...:

| 15 11 | 10 9 | 8 | 7 | 6 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| manufacturer specific | reserved | halt | Fault reset | Operation mode specific | Enable operation | Quick stop | Enable voltage | Switch on |

Bit0 ~ 3 和 Bit7：

The transmission of state machine is activated by the 5-bit correspondent control command.

| Command | Bit of the *controlword* | | | | | Transitions |
|---|---|---|---|---|---|---|
| | Fault reset | Enable operation | Quick stop | Enable voltage | Switch on | |
| Shutdown | 0 | X | 1 | 1 | 0 | 2,6,8 |
| Switch on | 0 | 0 | 1 | 1 | 1 | 3* |
| Switch on | 0 | 1 | 1 | 1 | 1 | 3** |
| Disable voltage | 0 | X | X | 0 | X | 7,9,10,12 |
| Quick stop | 0 | X | 0 | 1 | X | 7,10,11 |
| Disable operation | 0 | 0 | 1 | 1 | 1 | 5 |
| Enable operation | 0 | 1 | 1 | 1 | 1 | 4,16 |
| Fault reset | ↑ | X | X | X | X | 15 |

Device control command

;

Note:X means the bit could be ignored.

Bit4、5、6、8：

ese 4 bit has different definition under different control mode.

| Bit | Control mode | | |
|---|---|---|---|
| | profile position mode | profile velocity mode | homing mode |
| 4 | new_set_point | reserved | start_homeing_operation |
| 5 | change_set_immediatly | reserved | reserved |
| 6 | abs/rel | reserved | reserved |
| 8 | Halt | Halt | Halt |

Other bit： all are reserved bits.

## 6.2.2 statusword

| | |
|---|---|
| Index | 6041 h |
| Name | statusword |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RO |
| PDO Mapping | YES |
| Units | -- |
| Value Range | -- |
| Default Value | -- |

The bit instruction of statusword is as below ...

| bit | instruction |
|---|---|
| 0 | Ready to switch on |
| 1 | Switched on |
| 2 | Operation enabled |
| 3 | Fault |
| 4 | Voltage enabled |
| 5 | Quick stop |
| 6 | Switch on disabled |
| 7 | Warning |
| 9~8 | reserved |
| 10 | Target reached |
| 11 | Internal limit active |
| 13~12 | Operation mode specific |
| 15~14 | reserved |

Bit0 ~ 3 、Bit5 和 Bit6：

The combination of these bits indicates the state of the servo drive.

| Value (binary) | State |
|---|---|
| xxxx xxxx x0xx 0000 | Not ready to switch on |
| xxxx xxxx x1xx 0000 | Switch on disabled |
| xxxx xxxx x01x 0001 | Ready to switch on |
| xxxx xxxx x01x 0011 | Switched on |
| xxxx xxxx x01x 0111 | Operation enabled |
| xxxx xxxx x00x 0111 | Quick stop active |
| xxxx xxxx x0xx 1111 | Fault reaction active |
| xxxx xxxx x0xx 1000 | Fault |

Bit4： Voltage enabled

1 means the main circuit is powered on.

Bit5： Quick stop

0 means the servo drive will stop according to the settings（605A $_{h}$：quick_stop_option_code）

Bit7： Warning

1 means the servo drive could detect alarms.

Bit10： Target reached

In different control mode, the definition is different.

In profile position mode, when set position is reached, the bit will be set. After Halt is initiated and speed is reduced to 0, the bit will be set. When new position is settled, the bit will be cleared.

In Profile Velocity Mode, when the speed reaches the targeted speed, the bit will be set. After Halt is initiated and speed is reduced to 0, this bit will be set.

Bit11： Internal limit active

When the bit is 0, it means internal torque is bigger than the set value.

Bit12、13：

These 2 bits have different definition under different control mode.

| Bit | Control mode | | |
|---|---|---|---|
| | profile position mode | profile velocity mode | homing mode |
| 12 | Set-point acknowledge | Speed | Homing attained |
| 13 | Following error | Max slippage error | Homing error |

Other bits：

All reserved.

## 6.2.3 shutdown_option_code

When Operation Enable state transits to Ready to Switch On state，shutdown_option_code will determine how to stop the servo motor.

| | |
|---|---|
| Index | 605B $_{h}$ |
| Name | shutdown_option_code |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | -- |
| Value Range | 0 |
| Default Value | 0 |

| value | instruction |
|---|---|
| 0 | Excitation of servo motor is shut down and the servo motor will rotate freely till stop. |

## 6.2.4 disable_operation_option_code

When Operation Enable state transits to Switched On state，disable_operation_option_code will determine how to stop.

| | |
|---|---|
| Index | 605C $_{h}$ |
| Name | disable_operation_option_code |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | -- |
| Value Range | 0 |
| Default Value | 0 |

| value | instruction |
|---|---|
| 0 | Excitation of servo motor is shut down and the servo motor will rotate freely till stop. |

## 6.2.5 quick_stop_option_code

When Operation Enable state transits to Quick Reaction Active state，quick_stop_option_code will determine how to stop.

| | |
|---|---|
| Index | 605A $_{h}$ |
| Name | quick_stop_option_code |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | -- |
| Value Range | 6 |
| Default Value | 0 |

| value | instruction |
|---|---|
| 6 | When the servo motor decelerates urgently to still, QuickStop state is still kept. |

## 6.2.6 halt_option_code

When the bit8(halt) of the controlword is1 时，halt_option_code will determine how to stop..

| Index | 605D $_{h}$ |
|---|---|
| Name | halt_option_code |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | -- |
| Value Range | 2 |
| Default Value | 0 |

| value | instruction |
|---|---|
| 2 | Servo motor will decelerate urgently to still. |

## 6.2.7 fault_reaction_option_code

When alarms are detected, . fault_reaction_option_code will determine how to stop.

| Index | 605E $_{h}$ |
|---|---|
| Name | fault_reaction_option_code |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Units | -- |
| Value Range | 0 |
| Default Value | 0 |

| value | instruction |
|---|---|
| 0 | Excitation of servo motor is shut down and the servo motor will rotate freely till stop. |

# 7、Control mode

EDC servo drive currently supports 3 control modes in CANopen DSP402.

HOMING MODE

PROFILE VELOCITY MODE

PROFILE POSITION MODE

This chapter mainly describes these three control modes as above.

## 7.1 Parameters about control mode.

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 6060 h | VAR | modes_of_operation | INT8 | RW |
| 6061 h | VAR | modes_of_operation_display | INT8 | RO |

### 7.1.1 modes_of_operation

The control mode of the servo drive will be determined by the parameter modes_of_operation.

| | |
|---|---|
| Index | 6060 h |
| Name | modes_of_operation |
| Object Code | VAR |
| Data Type | INT8 |
| Access | RW |
| PDO Mapping | YES |
| Units | -- |
| Value Range | 1,3,6 |
| Default Value | 0 |

| Value | Instruction |
|---|---|
| 0 | NOP MODE |
| 1 | PROFILE POSITION MODE |
| 3 | PROFILE VELOCITY MODE |
| 6 | HOMING MODE |

## 7.1.2 modes_of_operation_display

The current control mode of the servo drive could be known by reading the parameter modes_of_operation_display

| | |
|---|---|
| Index | 6061 **h** |
| Name | modes_of_operation_display |
| Object Code | VAR |
| Data Type | INT8 |
| Access | RO |
| PDO Mapping | YES |
| Units | -- |
| Value Range | 1,3,6 |
| Default Value | 0 |

## 7.2 HOMING MODE

EDC servo drive currenly supports 4 kind of homing modes. The consumers need to choose appropriate and correspondent homing modes.

Users could set the homing methods, homing speed and acceleration speed. After the servo drive finds reference position, it could move the homing position toward and the moving distance is set by home_offset（607C $_{h}$）

### 7.2.1 Control word of homing mode.

| 15 ~ 9 | 8 | 7 ~ 5 | 4 | 3 ~ 0 |
|---|---|---|---|---|
| * | Halt | * | home_start_operation | * |

*: Please refer to the previous chapters

| Name | Value | Description |
|---|---|---|
| Homing operation start | 0 | Homing mode inactive |
| | 0 → 1 | Start homing mode |
| | 1 | Homing mode active |
| | 1 → 0 | Interrupt homing mode |
| Halt | 0 | Execute the instruction of bit 4 |
| | 1 | Stop axle with homing acceleration |

### 7.2.2 Status word of homing mode

| 15 ~ 14 | 13 | 12 | 11 | 10 | 9 ~ 0 |
|---|---|---|---|---|---|
| * | homing_error | homing_attained | * | target_reached | * |

*: Please refer to the previous chapters

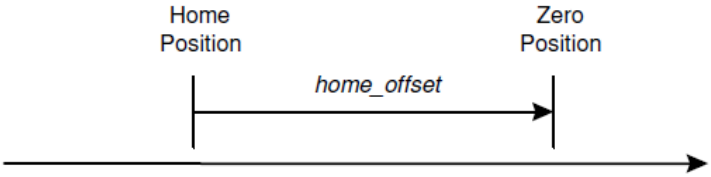
| Name | Value | Description |
|---|---|---|
| Target reached | 0 | Halt = 0: Home position not reached<br>Halt = 1: Axle decelerates |
| | 1 | Halt = 0: Home position reached<br>Halt = 1: Axle has velocity 0 |
| Homing attained | 0 | Homing mode not yet completed |
| | 1 | Homing mode carried out successfully |
| Homing error | 0 | No homing error |
| | 1 | Homing error occurred;<br>Homing mode carried out not successfully;<br>The error cause is found by reading the error code |

## 7.2.3 Parameters of the homing mode

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 607C h | VAR | home_offset | INT32 | RW |
| 6098 h | VAR | homing_method | INT8 | RW |
| 6099 h | ARRAY | homing_speeds | UINT32 | RW |
| 609A h | VAR | homing_acceleration | INT32 | RW |

**home_offset**

home_offset could set the distance between reference point and homing point.




| | |
|---|---|
| Index | 607C h |
| Name | home_offset |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | position units |
| Value Range | -- |
| Default Value | 0 |

**homing_method**

This parameters defines several kinds of homing methods.

| | |
|---|---|
| Index | 6098 h |
| Name | homing_method |
| Object Code | VAR |
| Data Type | INT8 |
| Access | RW |
| PDO Mapping | YES |
| Units | -- |
| Value Range | 3,4, 19,20 |
| Default Value | 1 |

Homing method form

| Method | direction | Target position | reference position | DS402 |
|---|---|---|---|---|
| 3 | Negative | Reference point switch | C pulse | 3 |
| 4 | Positive | Reference point switch | C pulse | 4 |
| 19 | Negative | Reference point switch | Reference point switch | 19 |
| 20 | Positive | Reference point switch | Reference point switch | 20 |

**homing_speeds**

There are 2 kinds of relevant speeds, homing speed and acceleration homing speed.

| Index | 6099 h |
|---|---|
| Name | homing_speeds |
| Object Code | ARRAY |
| No. of Elements | 2 |
| Data Type | INT32 |

| Sub-Index | 01 h |
|---|---|
| Name | speed_during_search_for_switch |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 0 |

| Sub-Index | 02 h |
|---|---|
| Name | speed_during_search_for_zero |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 0 |

**homing_acceleration**

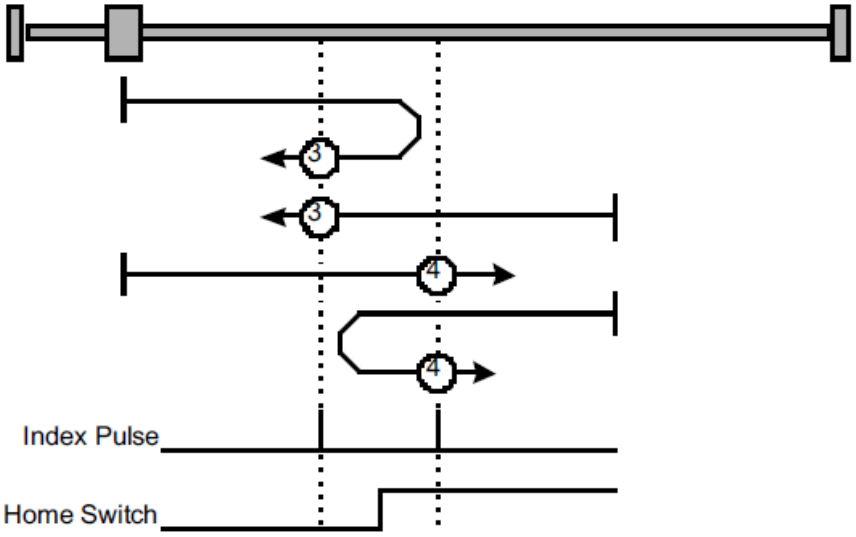
Homing_acceleration will define both the acceleration speed and deceleration speed at the process of homing.

| Index | 609A h |
|---|---|
| Name | homing_acceleration |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | acceleration units |
| Value Range | -- |
| Default Value | 0 |

## 7.2.4 homing method

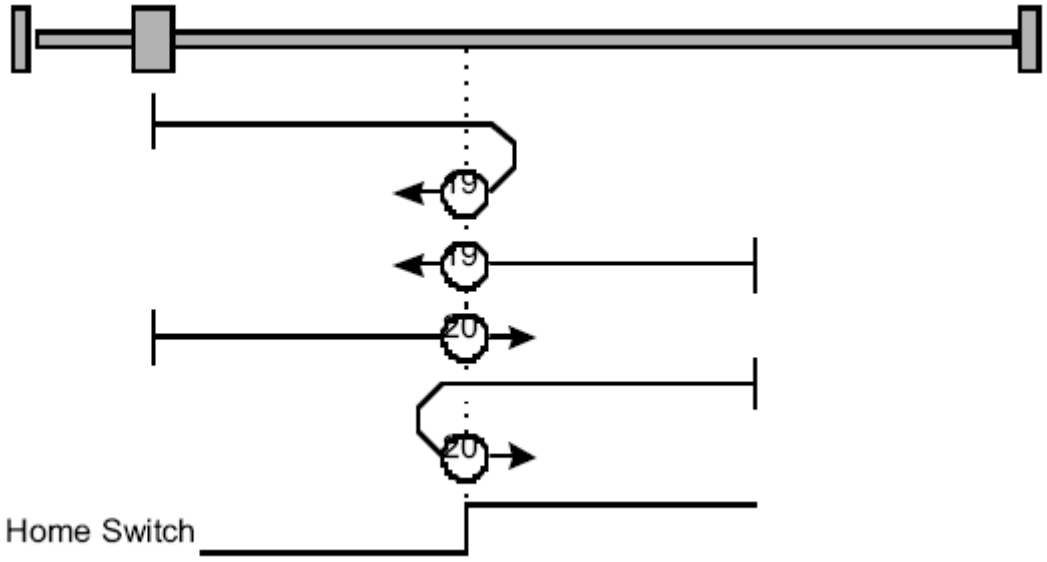
Method 3 and 4: **C pulse and reference point switch (ZPS signal)**

The initial moving direction of the servo drive relies on the state of reference point switch. Targeted homing position is on the left or right side of the reference point switch, one C pulse far away from the reference point switch.

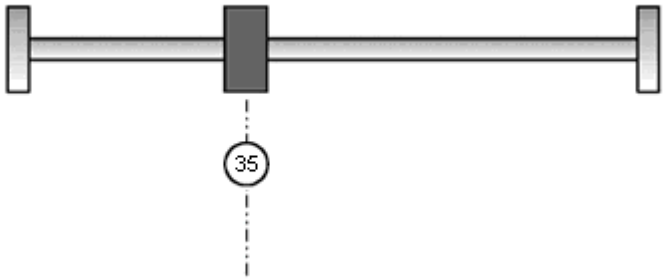



**Method 19 and 20 Reference point switch( ZPS signal)**

Method 19 and 20 only use Home Switch(ZPS) signal for homing. Homing method is very similar to method 3 and 4.



**Method 35** **set current position as the homing point.**



# 7.3 PROFILE VELOCITY MODE

## 7.3.1 control word of velocity mode.

| 15 ~ 9 | 8 | 7 ~ 4 | 3 ~ 0 |
|---|---|---|---|
| * | Halt | * | * |

*: Please refer to the previous chapters

| Name | Value | Description |
|---|---|---|
| Halt | 0 | Execute the motion |
| | 1 | Stop axle |

## 7.3.2 Status word of control mode

| 15 ~ 14 | 13 | 12 | 11 | 10 | 9 ~ 0 |
|---|---|---|---|---|---|
| * | MaxSlippageError | Speed | * | Target reached | * |

*: Please refer to the previous chapters

| Name | Value | Description |
|---|---|---|
| Target reached | 0 | Halt = 0: *Target velocity* not (yet) reached<br>Halt = 1: Axle decelerates |
| | 1 | Halt = 0: *Target velocity* reached<br>Halt = 1: Axle has velocity 0 |
| Speed | 0 | Speed is not equal 0 |
| | 1 | Speed is equal 0 |
| Max slippage error | 0 | Maximum slippage not reached |
| | 1 | Maximum slippage reached |

## 7.3.3 Parameters of speed control mode.

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 6069 $_h$ | VAR | velocity_sensor_actual_value | INT32 | RO |
| 606B $_h$ | VAR | velocity_demand_value | INT32 | RO |
| 606C $_h$ | VAR | velocity_actual_value | INT32 | RO |
| 609D $_h$ | VAR | velocity_window | UINT16 | RW |
| 606E $_h$ | VAR | velocity_window_time | UINT16 | RW |
| 606F $_h$ | VAR | velocity_threshold | UINT16 | RW |
| 6070 $_h$ | VAR | velocity_threshold_time | UINT16 | RW |
| 60FF $_h$ | VAR | target_velocity | INT32 | RW |

**velocity_sensor_actual_value**

The host could read velocity_sensor_actual_value to know the current rotation speed. The unit is internal speed unit.

| | |
|---|---|
| Index | 6069 **h** |
| Name | velocity_sensor_actual_value |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | 0.1rmps (1R/10min) |
| Value Range | -- |
| Default Value | -- |

**velocity_demand_value**

The host could read velocity_demand_value to know the set speed. The unit is speed unit of the customer.

| | |
|---|---|
| Index | 606B **h** |
| Name | velocity_demand_value |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RO |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | -- |

**velocity_actual_value**

The host could read velocity_actual_value to know the current speed. The unit is speed unit of the customer.

| | |
|---|---|
| Index | 606C **h** |
| Name | velocity_actual_value |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RO |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | -- |

**velocity_window**

The difference between Velocity_actual_value（606C $_{h}$）and target_velocity（60FF $_{h}$）is defined as the actual speed error window. If the actual speed error window is smaller than velocity_window（606D $_{h}$）during the time set by velocity_window_time（606E $_{h}$）, bit 10 of statusword (target_reached) will be set to indicate the speed has been reached.

| Index | 606D $_{h}$ |
|---|---|
| Name | velocity_window |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 20 R/10min |

**velocity_window_time**

velocity_window_time and velocity_window together form a speed window comparison tool.

| Index | 606E $_{h}$ |
|---|---|
| Name | velocity_window_time |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | ms |
| Value Range | -- |
| Default Value | 0 |

**velocity_threshold**

velocity_threshold indicates the range closed to the still to judge if the servo motor should stop.

| Index | 606F $_{h}$ |
|---|---|
| Name | velocity_threshold |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 10 R/10min |

**velocity_threshold_time**

velocity_threshold_time sets the minimum time during which the speed is below threshold velocity. The unit is ms. When the time during which the speed is below the threshold has surpassed the velocity_threshold_time,bit12 of the status word will be set as much as 1.

| | |
|---|---|
| Index | 6070 **h** |
| Name | velocity_threshold_time |
| Object Code | VAR |
| Data Type | UINT16 |
| Access | RW |
| PDO Mapping | YES |
| Units | ms |
| Value Range | -- |
| Default Value | 0 |

**target_velocity**

target_velocity is the targeted velocity.

| | |
|---|---|
| Index | 60FF **h** |
| Name | target_velocity |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 0 |

# 7.4 PROFILE POSITION MODE

## 7.4.1 control word for position mode.

| 15 ~ 9 | 8 | 7 | 6 | 5 | 4 | 3 ~ 0 |
|---|---|---|---|---|---|---|
| * | Halt | * | abs / rel | change set immediately | New set-point | * |

*: Please refer to the previous chapters.

| Name | Value | Description |
|---|---|---|
| New set-point | 0 | Does not assume *target position* |
| | 1 | Assume *target position* |
| Change set immediately | 0 | Finish the actual positioning and then start the next positioning |
| | 1 | Interrupt the actual positioning and start the next positioning |
| abs / rel | 0 | *Target position* is an absolute value |
| | 1 | *Target position* is a relative value |
| Halt | 0 | Execute positioning |
| | 1 | Stop axle with *profile deceleration* (if not supported with *profile acceleration*) |

## 7.4.2 Status word of position control mode.

| 15 ~ 14 | 13 | 12 | 11 | 10 | 9 ~ 0 |
|---|---|---|---|---|---|
| * | Following error | Set_point acknowledge | * | Target reached | * |

*: Please refer to previous chapters.

| Name | Value | Description |
|---|---|---|
| Target reached | 0 | Halt = 0: *Target position* not reached<br>Halt = 1: Axle decelerates |
| | 1 | Halt = 0: *Target position* reached<br>Halt = 1: Velocity of axle is 0 |
| Set-point acknowledge | 0 | Trajectory generator has not assumed the positioning values (yet) |
| | 1 | Trajectory generator has assumed the positioning values |
| Following error | 0 | No following error |
| | 1 | Following error |

## 7.4.3 Parameters about position control

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 607A $_{h}$ | VAR | target_position | INT32 | RW |
| 6081 $_{h}$ | VAR | profile_velocity | UINT32 | RW |
| 6083 $_{h}$ | VAR | profile_acceleration | UINT32 | RW |
| 6084 $_{h}$ | VAR | profile_deceleration | UINT32 | RW |
| 6085 $_{h}$ | VAR | quick_stop_deceleration | UINT32 | RW |

**target_position**

Target_position is targeted position, which could be a relative value or a absolute value. It is up to bit6 f the control word.

| | |
|---|---|
| Index | 607A $_{h}$ |
| Name | target_ position |
| Object Code | VAR |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | position units |
| Value Range | -- |
| Default Value | 0 |

**profile_velocity**

Profile_velocity means the speed that could be reached through acceleration after the positioning is initialized.

| | |
|---|---|
| Index | 6081 $_{h}$ |
| Name | profile_velocity |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | speed units |
| Value Range | -- |
| Default Value | 0 |

**profile_acceleration**

profile_acceleration is the acceleration speed before reaching the set position.

| | |
|---|---|
| Index | 6083 **h** |
| Name | profile_acceleration |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | acceleration units |
| Value Range | -- |
| Default Value | 0 R/10min/s |

**profile_deceleration**

profile_deceleration is the deceleration speed before reaching the set position.

| | |
|---|---|
| Index | 6084 **h** |
| Name | profile_deceleration |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | acceleration units |
| Value Range | -- |
| Default Value | 0 R/10min/s |

**quick_stop_deceleration**

quick_stop_deceleration is the deceleration speed when Quick Stop happens.

| | |
|---|---|
| Index | 6085 **h** |
| Name | quick_stop_deceleration |
| Object Code | VAR |
| Data Type | UINT32 |
| Access | RW |
| PDO Mapping | YES |
| Units | acceleration units |
| Value Range | -- |
| Default Value | 0 R/10min/s |

## 7.4.4 Function descreption

There are two ways to reach targeted position.

Single step setting

After the servo motor reaches the target position, the servo drive will notify the host controller “target position reached”. And then, the servo drive will obtain new target position and start movement. Ahead of obtaining new target position, normally the speed of the servo motor will keep still.

Continuous setting：After the servo motor reaches the target position, it will move forward to next previously set target position. Then it could keep moving without any pause between 2 target positions and no speed reduction is necessary.

Both of the two methods above could be changed by bit 4, bit 5 of the control word and bit 12 （set_point_acknowledge） of the status word. Through handshake mechanism, position control that is being executed could be terminated and re-established through these bits.

Procedure of single step setting:

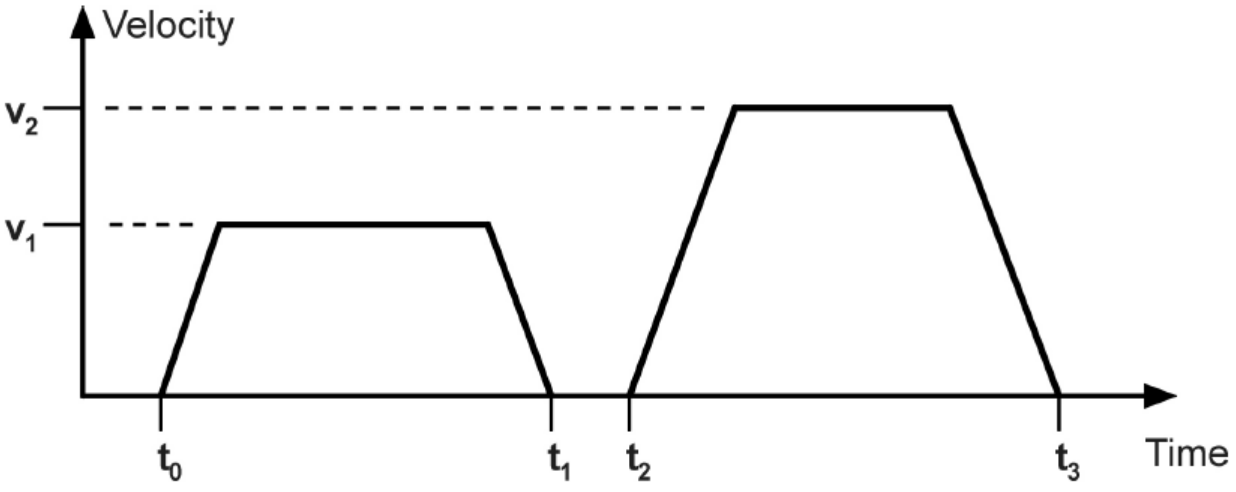
At first, setting NMT as operational and set control mode parameter （6060 $_{h}$）as 1.

1、Set target position (target_ position ：607A $_{h}$）and other parameters according to the requirements of actual application.

2、set bit4 of control word（new_set_point）as 1 . Set bit 5 (change_set_immediately) as o. set bit 6 ( aboslue/relative) according to the type of targeted position.

3、Set bit12（set_point_acknowledge）of the status word about device response and then execute the position control.

4、After reaching the targeted position, the servo drive will respond through bit 10 of (target_reached). And then it will follow the program to keep moving or accept new target position.




The procedure of continuous setting

At first set NMT as Operational and set control mode parameter （6060 $_{h}$）as 1.

1、Set the first target position（target_ position ：607A $_{h}$）, target speed, acceleration/deceleration and relative parameters.

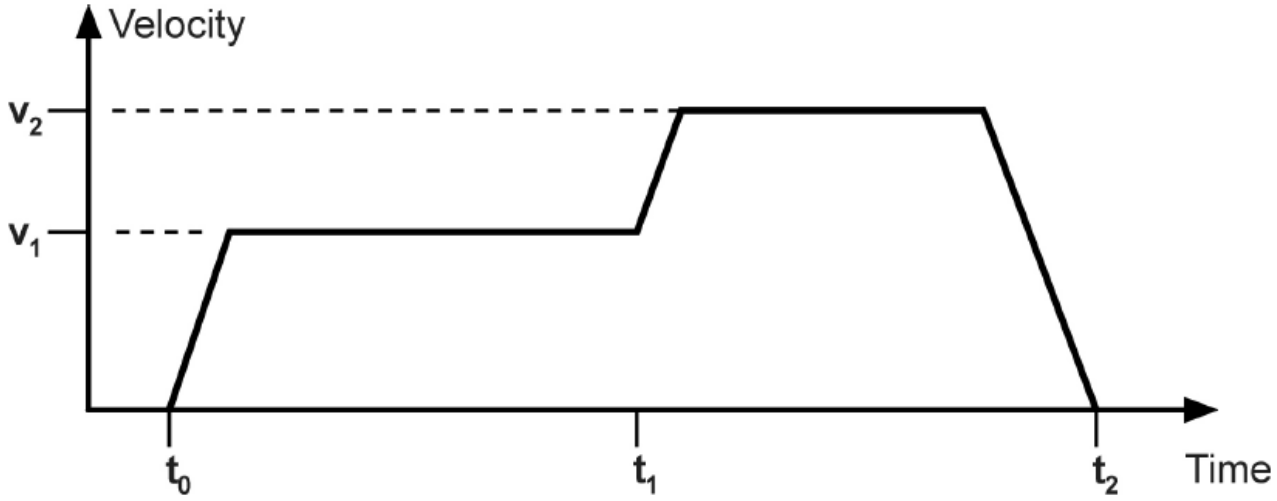
2、Set bit4 (new_set_point）of the control word as 1. Set bit 5（change_set_immediately）as 0. Set bit6 (absolute/relative) according to the type of target position (absolute/relative).

3、 Set bit12 of status word (set_point_acknowledge）for servo drive response and then execute position control.

4、 Set the second target position（target_ position ：607A $_h$）,objective speed, acceleration/deceleration speed and relative parameters.

5、 Set bit4 (new_set_point) of the control word as 1. set bit5(change_set_immediately) as 0. Set bit6 (absolute/relative) according to the type of target position (absolute/relative).

6、 After reaching the first target, the servo drive will keep moving forward to the second target position. After reaching the second target position, the servo drive will respond through bit 10 (target_reached) of status word. And it will follow the program to keep moving or accept new targeted position.



## 7.5 interpolation position mode

### 7.5.1 Control word of interpolation position mode

| 15 ~ 9 | 8 | 7 | 6 | 5 | 4 | 3 ~ 0 |
|---|---|---|---|---|---|---|
| * | Halt | * | * | * | Enable ip mode | * |

*: Please refer to the chapters ahead

| Name | Value | Description |
|---|---|---|
| Enable ip mode | 0 | Interpolated position mode inactive |
| | 1 | Interpolated position mode active |
| Halt | 0 | Execute the instruction of bit 4 |
| | 1 | Stop axle |

### 7.5.2 Status word of interpolation position mode

| 15 ~ 14 | 13 | 12 | 11 | 10 | 9 ~ 0 |
|---|---|---|---|---|---|
| * | * | ip mode active | * | Target reached | * |

*: Please refer to the chapters ahead

| Name | Value | Description |
|---|---|---|
| Target reached | 0 | Halt = 0: Position not (yet) reached<br>Halt = 1: Axle decelerates |
| | 1 | Halt = 0: Position reached<br>Halt = 1: Axle has velocity 0 |
| ip mode active | 0 | Interpolated position mode inactive |
| | 1 | Interpolated position mode active |

### 7.5.3 Parameters of position interpolation control

| Index | Object | Name | Type | Attr. |
|---|---|---|---|---|
| 60C0 $_h$ | VAR | Interpolation sub mode select | INT16 | RW |
| 60C1 $_h$ | ARRAY | Interpolation data record | INT32 | RW |

### Interpolation sub mode select

Interpolation sub mode select is used to select the method of interpolation under IP control. EDC servo drive only offers linear interpolation.

| | |
|---|---|
| Index | 60C0h |
| Name | Interpolation sub mode select |
| Object Code | VAR |
| Data Type | INT16 |
| Access | RW |
| PDO Mapping | NO |
| Value Range | 0 |
| Default Value | 0 |
| Comment | 0: Linear interpolation |

### Interpolation data record

Interpolation data record is used to reserve interpolation potion data. EDC servo drive's interpolation command only uses the first data whose subindex is 1.

| | |
|---|---|
| Index | 60C1h |
| Subindex | 0 |
| Object Code | ARRAY |
| Data Type | INT32 |
| Access | RO |
| PDO Mapping | YES |
| Value Range | INT8 |
| Default Value | 2 |
| Comment | number of entries |

| | |
|---|---|
| Index | 60C1h |
| Subindex | 1 |
| Object Code | ARRAY |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Value Range | INT32 |
| Default Value | 0 |
| Comment | the first parameter of ip function |

| | |
|---|---|
| Index | 60C1h |
| Subindex | 2 |
| Object Code | ARRAY |
| Data Type | INT32 |
| Access | RW |
| PDO Mapping | YES |
| Value Range | INT32 |
| Default Value | 0 |
| Comment | The second parameter of ip function |

## 7.5.4 Function description

Some hints:

1. In IP mode, the host should at first set the servo's PDO receiving method into sync mode ( Use SYNC frame to receive and send synchronization information). Because SYNC is broad casted, every servo drive will only update PDO data after receiving this signal.
2. Before SYNC is sent, we need host to send position data Xi and control word to the servo drive.
3. Position data buffer is not given by servo drive to avoid delay.
4. When there is sync signal delay, servo drive will use the last sync date to do interpolation.
5. When sync signal delay is reached 2 times of the sync period, interpolation cycle overtime alarm will happen. And then servo drive will stop.

Recommended RPDO configuration:

When you use only one RPDO,

| Control word(index:6040h,subindex:0h) | 32bit position reference (index:60C1h,subindex:01h) |
|---|---|

When you use two RPDO,

| Control word(index:6040h,subindex:0h) |
|---|

| 32bit position reference (index:60C1h,subindex:01h) |
|---|

Configuration process:

1. Configure PDO dynamically. (RPDO1 is configured as index: 6040h, subindex: 0h, RPDO2 is configured as index 60c1h, subindex: 1h)
2. Set sync cycle time (1006h), the unit is micro send (us)
3. Set PDO as Sync mode (Set the object dictionary (index: 1400h, subindex: 02h) as 1. Set object dictionary (index: 1401h, subindex: 02h) as 1). If sending PDO needs to be in sync mode as well, we need to set object dictionary (index: 1800h, subindex: 02h) as 1 and (index: 6060h, subindex: 0h) as 1 as well.
4. Set control mode as position interpolation mode. (Set object dictionary (index: 6060h, subindex: 0h) as 7).
5. Reset the communication and then reactivate the communication.

# 8、CAN communication parameters

CANopen parameters of EDC servo drive.

| Parameter number | Name | Re-power on required | Function and instruction |
|---|---|---|---|
| Pn063 | **Axis address** | yes | Axis address of CANopen communication. When the ID on the drive's front panel was set as F, this parameter will be used as axis address. When the ID is not F, ID on the front panel will be used as axis address. |
| Pn064 | **Communication speed** | yes | CANopen communication baudrate<br>[0] 50Kbps<br>[1] 100Kbps<br>[2] 125Kbps<br>[3] 250Kbps<br>[4] 500Kbps<br>[5] 1Mbps |
| Pn065 | **CAN communication enable** | yes | CANopen enabled. |

# 9、 CAN communication example

The entire test below is based on two conditions …
1. Communication has been established correctly.
2. The address of the servo drive is 1.

If EDC software version >3.10, the EDC servo drive has two receive PDO （RPDO1：6040，RPDO2：607A/6081）, and two send PDO (TPDO1：6041，TPDO2：6064/606C） by default.

If EDC software version ≤3.10, the EDC servo drive has one receive PDO （RPDO1：6040/60FF）, and one send PDO (TPDO1：6041/6064） by default.

## 9.1 SDO configuration

SDO operation is to read and write parameters (06001→ host sends 0581-→slave sends)
Address: 0x3022（Pn034）. Write 1000. And then read this parameter.
Activate the downloading process: 2B，3022，00，FC18
That is …
601 2B 22 30 00 18 FC 00 00
The servo drive should respond 60，3022，00，00，00，00，00

That is 581 60 22 30 00 00 00 00 00

Activate the uploading: 40，3022，00，0000

That is 601 40 22 30 00 00 00 00 00

The servo drive needs to respond: 43，3022，00，FC18
That is: 581 43 22 30 00 18 FC 00 00

## 9.2 PDO Configuration

// pulse，Speed 0.1rpm

Example: To configure two RPDO, one of which is 6040h and the other are 607A and 6081h)
RPDO MAPPing
601 2F 00 16 00 00 00 00 00 //RPDO1 stop
first RPDO 201
601 23 00 16 01 10 00 40 60 //6040h
601 2F 00 16 00 01 00 00 00 // RPDO1 enable

601 2F 01 16 00 00 00 00 00 //RPDO2 stop
Second RPDO 301
601 23 01 16 01 20 00 7A 60 //607Ah and 6081h

601 23 01 16 02 20 00 81 60
601 2F 01 16 00 02 00 00 00// RPDO2 enable

And then set the transmit PDO as SYNC or Timing method. The default setting is Time method.

After configuring the PDO, if you need to activate the configuration, you need to reset the communication.
NMT management: 00 82 01// Reset the servo drive with the axis address as much as 1.

Reactivate the communication.
00 01 01

Attention：
1) Before configuration, please stop PDO. For example, Cleaning the value with index 1600h and sub-index 00, cleaning the value to 0 is necessary). After configuration, please set a correct number of PDO( For example, set the value with index 1600h and sub-index 00 as 1) to activate the PDO.
2) Please pay attention to the data length and number. Wrong setting will lead to wrong configuration.
3) After configuration, resetting communication is necessary to activate the PDO.

## 9.3 Profile Position Mode

At first, please configure PDO according to the example above and activate the communication.
And then, please set the control mode.
601 2F 60 60 00 01 00 00 00//set 6060h as 1（position contrl is PP）
And then, set status machine
601 2B 40 60 00 06 00 00 00//set 6040h as 6
601 2B 40 60 00 07 00 00 00 //set 6040h as 7
601 2B 40 60 00 0F 00 00 00 //set 6040h as F，servo-on；

And then, send data by PDO
Let servo motor rotate for 5 revolutions ( Set PDO1 as 6040(status word), PDO2 as 607A(position pulse number) and 6081(velocity, unit as much as 0.1rpm)

Send RPDO2 The data is as below …
301 50 C3 00 00 2C 01 00 00(50000,300)// 50 C3 00 00 is position data，that is, 50000 pulses；2C 01 00
00 is speed，that is, 30rpm；

Send RPDO1 as below
1、201 0F 00 //； Clear the bit4 of 6040 as 0.
2、201 1F 00 // Clear the bit4 of 6040 as 1 and servo motor is operating under absolute position; Motor runs.
3、201 0F 00 //Clear the bit4 of 6040.
4、201 5F 00 // Clear the bit4 of 6040 as 1. The servo motor runs under incremental position.

5、201 0F 00 //Clear bit4 of 6040 as 0.

Attention:
1) The servo drive is using ↑of 6040's bit 4 to accept new position order. So after every single operation, the bit needs to be cleared. Host needs to check bit12 of status word 6040 in the servo drive to decide whether or not to give new data to servo systems. When status word 6041 in the servo drives 0, it means the servo drive is ready for new data and order. If the value is 1, the order won't be executed even if there is data for the servo drive to receive.
2) In absolute approach, continuous position updating is required.

If you want to change the operating distance, you need to send RPDO2 again.
RPDO2:
301 B0 3C FF FF 2C 01 00 00 (-50000,-300)//That is, -50000 pulses; 30rpm.

## 9.4 Interplate Position Mode

At first, configure PDO
/receive 2 PDO by default：RPDO1：6040 RPDO2：60C1,sub01
// Send 2 PDO by default：TPDO1：6041 TPDO2：6064/606C
// pulse，Velocity 0.1rpm

Configure 2 RPDO，RPDO1：6040h　RPDO2：60C1h, sub01
RPDO MAPPing
601 2F 00 16 00 00 00 00 00 //RPDO1 stop
first RPDO 201
601 23 00 16 01 10 00 40 60 //6040h
601 2F 00 16 00 01 00 00 00 // RPDO1 enable

601 2F 01 16 00 00 00 00 00 //RPDO2 stop
Second RPDO 301
601 23 01 16 01 20 01 C1 60 //60C1h,sub01
601 2F 01 16 00 01 00 00 00// RPDO2 enable

Configure 2 TPDO，TPDO1：6041h TPDO2：6064h/606Ch
RPDO MAPPing
601 2F 00 1A 00 00 00 00 00 //TPDO1 stop
first RPDO 181
601 23 00 1A 01 10 00 41 60 //6041h
601 2F 00 1A 00 01 00 00 00 // TPDO1 enable

601 2F 01 1A 00 00 00 00 00 //RPDO2 stop
Second RPDO 281
601 23 01 1A 01 20 00 64 60 //6064h and 606Ch
601 23 01 1A 02 20 00 6C 60 //
601 2F 01 1A 00 02 00 00 00// TPDO2 enable

Set Sync time.
601 23 06 10 00 E8 03 00 00 //1006h----->1000us

Set interpolate time
601 2F C2 60 01 10 00 00 00 //60C2h----->1ms
601 2F C2 60 02 FD 00 00 00 //

Configure the PDO receiving and sending both by the means of the sync step and sync frame.
Set 1400h
601 2F 00 14 02 01 00 00 00 //1400----SYNC
Set 1401h
601 2F 01 14 02 01 00 00 00 //1401----SYNC
Set 1800h
601 2F 00 18 02 01 00 00 00 //1800----SYNC
Set 1801h
601 2F 01 18 02 01 00 00 00 //1801----SYNC

Reset the communication to active dynamic PDO configuration.
00 82 01 //reset communication

Set control mode
601 2F 60 60 00 07 00 00 00// ( IP position control)
And then, set the status machine
601 2B 40 60 00 06 00 00 00//Set 6040h as 6
601 2B 40 60 00 07 00 00 00 //Set 6040h as 7
601 2B 40 60 00 0F 00 00 00 // Set 6040h as F to servo on.

Activate the communicaiton
00 01 01
The host sends signals by the period of 1000us.
301 10 00 00 00 //16 pulses
201 1F 00 //IP
80 periodical sending

## 9.5 Homing

Set the control mode as homing control.
601 2F 60 60 00 06 00 00 00// Set the control mode as homing control.

601 2F 98 60 00 04 00 00 00//Use the fourth way to set the homing mode.

Set the status machine
601 2B 40 60 00 06 00 00 00
601 2B 40 60 00 07 00 00 00
601 2B 40 60 00 0F 00 00 00 //Servo On

Send data through PDO. ( Set PDO1 as 6040( status word). Set PDO2 as 607A( Position pulse number) and 6081. ( Speed, unit 0.1rpm)

Set the homing method as 10rpm.
601 23 99 60 02 64 00 00 00

Homing is started.
201 1F 00

Cancel homing.
201 0F 0

# Appendix Object dictionary

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 1000 | -- | VAR | device_type | UINT32 | RO | NO | ● | | | | |
| 1001 | -- | VAR | error_register | UINT8 | RO | NO | ● | | | | |
| 1003 | -- | VAR | pre_defined_error_field | UINT8 | RW | NO | ● | | | | |
| 1005 | -- | VAR | cob_id_sync | UINT32 | RW | NO | ● | | | | |
| 1006 | -- | VAR | communication_cycle_period | UINT32 | RW | NO | ● | | | | |
| 1007 | -- | VAR | synchronous_window_length | UINT32 | RW | NO | ● | | | | |
| 1008 | -- | VAR | manufacturer_device_name | STR | RO | NO | ● | | | | |
| 1009 | -- | VAR | manufacturer_hardware_version | STR | RO | NO | ● | | | | |
| 100A | -- | VAR | manufacturer_software_version | STR | RO | NO | ● | | | | |
| 1014 | -- | VAR | cob_id_emergency_message | UINT32 | RW | NO | ● | | | | |
| 1016 | -- | ARRAY | consumer_heartbeat_time | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | consumer_heartbeat_time1 | UINT32 | RW | NO | ● | | | | |
| 1017 | | VAR | producer_heartbeat_time | UINT16 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 1400 | -- | RECORD | receive_pdo_parameter_rpdo1 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_rpdo1 | UINT8 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_rpdo1 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_rpdo1 | UINT8 | RW | NO | ● | | | | |
| 1401 | -- | RECORD | receive_pdo_parameter_rpdo2 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_rpdo2 | UINT8 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_rpdo2 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_rpdo2 | UINT8 | RW | NO | ● | | | | |
| 1402 | -- | RECORD | receive_pdo_parameter_rpdo3 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_rpdo3 | UINT8 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_rpdo3 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_rpdo3 | UINT8 | RW | NO | ● | | | | |
| 1403 | -- | RECORD | receive_pdo_parameter_rpdo4 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_rpdo4 | UINT8 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_rpdo4 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_rpdo4 | UINT8 | RW | NO | ● | | | | |
| 1600 | -- | RECORD | receive_pdo_mapping_rpdo1 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_rpdo1 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_rpdo1 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_rpdo1 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_rpdo1 | UINT32 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 1601 | -- | RECORD | receive_pdo_mapping_rpdo2 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_rpdo2 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_rpdo2 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_rpdo2 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_rpdo2 | UINT32 | RW | NO | ● | | | | |
| 1602 | -- | RECORD | receive_pdo_mapping_rpdo3 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_rpdo3 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_rpdo3 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_rpdo3 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_rpdo3 | UINT32 | RW | NO | ● | | | | |
| 1603 | -- | RECORD | receive_pdo_mapping_rpdo4 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_rpdo4 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_rpdo4 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_rpdo4 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_rpdo4 | UINT32 | RW | NO | ● | | | | |
| 1800 | -- | RECORD | transmit_pdo_parameter_tpdo1 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_tpdo1 | UINT32 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_tpdo1 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_tpdo1 | UINT8 | RW | NO | ● | | | | |
| | 3 | | inhibit_time_tpdo1 | UINT16 | RW | NO | ● | | | | |
| | 5 | | event_timer_tpdo1 | UINT16 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 1801 | -- | RECORD | transmit_pdo_parameter_tpdo2 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_tpdo2 | UINT32 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_tpdo2 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_tpdo2 | UINT8 | RW | NO | ● | | | | |
| | 3 | | inhibit_time_tpdo2 | UINT16 | RW | NO | ● | | | | |
| | 5 | | event_timer_tpdo2 | UINT16 | RW | NO | ● | | | | |
| 1802 | -- | RECORD | transmit_pdo_parameter_tpdo3 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_tpdo3 | UINT32 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_tpdo3 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_tpdo3 | UINT8 | RW | NO | ● | | | | |
| | 3 | | inhibit_time_tpdo3 | UINT16 | RW | NO | ● | | | | |
| | 5 | | event_timer_tpdo3 | UINT16 | RW | NO | ● | | | | |
| 1803 | -- | RECORD | transmit_pdo_parameter_tpdo4 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries_tpdo4 | UINT32 | RO | NO | ● | | | | |
| | 1 | | cob_id_used_by_pdo_tpdo4 | UINT32 | RO | NO | ● | | | | |
| | 2 | | transmission_type_tpdo4 | UINT8 | RW | NO | ● | | | | |
| | 3 | | inhibit_time_tpdo4 | UINT16 | RW | NO | ● | | | | |
| | 5 | | event_timer_tpdo4 | UINT16 | RW | NO | ● | | | | |
| 1A00 | -- | RECORD | transmit_pdo_mapping_tpdo1 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_tpdo1 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_tpdo1 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_tpdo1 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_tpdo1 | UINT32 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 1A01 | -- | RECORD | transmit_pdo_mapping_tpdo2 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_tpdo2 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_tpdo2 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_tpdo2 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_tpdo2 | UINT32 | RW | NO | ● | | | | |
| 1A02 | -- | RECORD | transmit_pdo_mapping_tpdo3 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_tpdo3 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_tpdo3 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_tpdo3 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_tpdo3 | UINT32 | RW | NO | ● | | | | |
| 1A03 | -- | RECORD | transmit_pdo_mapping_tpdo4 | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | ● | | | | |
| | 1 | | first_mapped_object_tpdo4 | UINT32 | RW | NO | ● | | | | |
| | 2 | | second_mapped_object_tpdo4 | UINT32 | RW | NO | ● | | | | |
| | 3 | | third_mapped_object_tpdo4 | UINT32 | RW | NO | ● | | | | |
| | 4 | | fourth_mapped_object_tpdo4 | UINT32 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | support | | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 3000 | -- | VAR | Correspondent to Pn000 | UINT16 | RW | NO | ● | | | | | |
| 3001 | -- | VAR | Correspondent to Pn001 | UINT16 | RW | NO | ● | | | | | |
| 3002 | -- | VAR | Correspondent to Pn002 | UINT16 | RW | NO | ● | | | | | |
| 3003 | -- | VAR | Correspondent to Pn003 | UINT16 | RW | NO | ● | | | | | |
| 3004 | -- | VAR | Correspondent to Pn004 | UINT16 | RW | NO | ● | | | | | |
| 3005 | -- | VAR | Correspondent to Pn005 | UINT16 | RW | NO | ● | | | | | |
| 3006 | -- | VAR | Correspondent to Pn006 | UINT16 | RW | NO | ● | | | | | |
| 3007 | -- | VAR | Correspondent to Pn007 | UINT16 | RW | NO | ● | | | | | |
| 3008 | -- | VAR | Correspondent to Pn008 | UINT16 | RW | NO | ● | | | | | |
| 3009 | -- | VAR | Correspondent to Pn009 | UINT16 | RW | NO | ● | | | | | |
| 300A | -- | VAR | Correspondent to Pn010 | UINT16 | RW | NO | ● | | | | | |
| 300B | -- | VAR | Correspondent to Pn011 | UINT16 | RW | NO | ● | | | | | |
| 300C | -- | VAR | Correspondent to Pn012 | UINT16 | RW | NO | ● | | | | | |
| 300D | -- | VAR | Correspondent to Pn013 | UINT16 | RW | NO | ● | | | | | |
| 300E | -- | VAR | Correspondent to Pn014 | UINT16 | RW | NO | ● | | | | | |
| 300F | -- | VAR | Correspondent to Pn015 | UINT16 | RW | NO | ● | | | | | |
| 3010 | -- | VAR | Correspondent to Pn016 | UINT16 | RW | NO | ● | | | | | |
| 3011 | -- | VAR | Correspondent to Pn017 | UINT16 | RW | NO | ● | | | | | |
| 3012 | -- | VAR | Correspondent to Pn018 | UINT16 | RW | NO | ● | | | | | |
| 3013 | -- | VAR | Correspondent to Pn019 | UINT16 | RW | NO | ● | | | | | |
| 3014 | -- | VAR | Correspondent to Pn020 | UINT16 | RW | NO | ● | | | | | |
| 3015 | -- | VAR | Correspondent to Pn021 | UINT16 | RW | NO | ● | | | | | |
| 3016 | | VAR | Correspondent to Pn022 | UINT16 | RW | NO | ● | | | | | |
| 3017 | -- | VAR | Correspondent to Pn023 | UINT16 | RW | NO | ● | | | | | |
| 3018 | -- | VAR | Correspondent to Pn024 | UINT16 | RW | NO | ● | | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | Support | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 3019 | -- | VAR | Correspondent to Pn025 | UINT16 | RW | NO | ● | | | | | |
| 301A | -- | VAR | Correspondent to Pn026 | UINT16 | RW | NO | ● | | | | | |
| 301B | -- | VAR | Correspondent to Pn027 | UINT16 | RW | NO | ● | | | | | |
| 301C | -- | VAR | Correspondent to Pn028 | UINT16 | RW | NO | ● | | | | | |
| 301D | -- | VAR | Correspondent to Pn029 | UINT16 | RW | NO | ● | | | | | |
| 301E | -- | VAR | Correspondent to Pn030 | UINT16 | RW | NO | ● | | | | | |
| 301F | -- | VAR | Correspondent to Pn031 | UINT16 | RW | NO | ● | | | | | |
| 3020 | -- | VAR | Correspondent to Pn032 | UINT16 | RW | NO | ● | | | | | |
| 3021 | -- | VAR | Correspondent to Pn033 | UINT16 | RW | NO | ● | | | | | |
| 3022 | -- | VAR | Correspondent to Pn034 | UINT16 | RW | NO | ● | | | | | |
| 3023 | -- | VAR | Correspondent to Pn035 | UINT16 | RW | NO | ● | | | | | |
| 3024 | -- | VAR | Correspondent to Pn036 | UINT16 | RW | NO | ● | | | | | |
| 3025 | -- | VAR | Correspondent to Pn037 | UINT16 | RW | NO | ● | | | | | |
| 3026 | -- | VAR | Correspondent to Pn038 | UINT16 | RW | NO | ● | | | | | |
| 3027 | -- | VAR | Correspondent to Pn039 | UINT16 | RW | NO | ● | | | | | |
| 3028 | -- | VAR | Correspondent to Pn040 | UINT16 | RW | NO | ● | | | | | |
| 3029 | -- | VAR | Correspondent to Pn041 | UINT16 | RW | NO | ● | | | | | |
| 302A | -- | VAR | Correspondent to Pn042 | UINT16 | RW | NO | ● | | | | | |
| 302B | -- | VAR | Correspondent to Pn043 | UINT16 | RW | NO | ● | | | | | |
| 302C | -- | VAR | Correspondent to Pn044 | UINT16 | RW | NO | ● | | | | | |
| 302D | -- | VAR | Correspondent to Pn045 | UINT16 | RW | NO | ● | | | | | |
| 302E | -- | VAR | Correspondent to Pn046 | UINT16 | RW | NO | ● | | | | | |
| 302F | | VAR | Correspondent to Pn047 | UINT16 | RW | NO | ● | | | | | |
| 3030 | -- | VAR | Correspondent to Pn048 | INT16 | RW | NO | ● | | | | | |
| 3031 | -- | VAR | Correspondent to Pn049 | UINT16 | RW | NO | ● | | | | | |
| 3032 | -- | VAR | Correspondent to Pn050 | UINT16 | RW | NO | ● | | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | Support | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 3033 | -- | VAR | Correspondent to Pn051 | UINT16 | RW | NO | ● | | | | | |
| 3034 | -- | VAR | Correspondent to Pn052 | UINT16 | RW | NO | ● | | | | | |
| 3035 | -- | VAR | Correspondent to Pn053 | UINT16 | RW | NO | ● | | | | | |
| 3036 | -- | VAR | Correspondent to Pn054 | UINT16 | RW | NO | ● | | | | | |
| 3037 | -- | VAR | Correspondent to Pn055 | UINT16 | RW | NO | ● | | | | | |
| 3038 | -- | VAR | Correspondent to Pn056 | UINT16 | RW | NO | ● | | | | | |
| 3039 | -- | VAR | Correspondent to Pn057 | UINT16 | RW | NO | ● | | | | | |
| 303A | -- | VAR | Correspondent to Pn058 | UINT16 | RW | NO | ● | | | | | |
| 303B | -- | VAR | Correspondent to Pn059 | UINT16 | RW | NO | ● | | | | | |
| 303C | -- | VAR | Correspondent to Pn060 | UINT16 | RW | NO | ● | | | | | |
| 303D | -- | VAR | Correspondent to Pn061 | UINT16 | RW | NO | ● | | | | | |
| 303E | -- | VAR | Correspondent to Pn062 | UINT16 | RW | NO | ● | | | | | |
| 303F | -- | VAR | Correspondent to Pn063 | UINT16 | RW | NO | ● | | | | | |
| 3040 | -- | VAR | Correspondent to Pn064 | UINT16 | RW | NO | ● | | | | | |
| 3041 | -- | VAR | Correspondent to Pn065 | UINT16 | RW | NO | ● | | | | | |
| 3042 | -- | VAR | Correspondent to Pn066 | UINT16 | RW | NO | ● | | | | | |
| 3043 | -- | VAR | Correspondent to Pn067 | UINT16 | RW | NO | ● | | | | | |
| 3044 | -- | VAR | Correspondent to Pn068 | UINT16 | RW | NO | ● | | | | | |
| 3045 | -- | VAR | Correspondent to Pn069 | UINT16 | RW | NO | ● | | | | | |
| 3046 | -- | VAR | Correspondent to Pn070 | UINT16 | RW | NO | ● | | | | | |
| 3047 | -- | VAR | Correspondent to Pn071 | UINT16 | RW | NO | ● | | | | | |
| 3048 | -- | VAR | Correspondent to Pn072 | UINT16 | RW | NO | ● | | | | | |
| 3049 | | VAR | Correspondent to Pn073 | UINT16 | RW | NO | ● | | | | | |
| 304A | -- | VAR | Correspondent to Pn074 | UINT16 | RW | NO | ● | | | | | |
| 304B | -- | VAR | Correspondent to Pn075 | UINT16 | RW | NO | ● | | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | Support | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 304C | -- | VAR | Correspondent to Pn076 | UINT16 | RW | NO | ● | | | | | |
| 304D | -- | VAR | Correspondent to Pn077 | UINT16 | RW | NO | ● | | | | | |
| 304E | -- | VAR | Correspondent to Pn078 | UINT16 | RW | NO | ● | | | | | |
| 304F | -- | VAR | Correspondent to Pn079 | UINT16 | RW | NO | ● | | | | | |
| 3050 | -- | VAR | Correspondent to Pn080 | INT16 | RW | NO | ● | | | | | |
| 3051 | -- | VAR | Correspondent to Pn081 | INT16 | RW | NO | ● | | | | | |
| 3052 | -- | VAR | Correspondent to Pn082 | INT16 | RW | NO | ● | | | | | |
| 3053 | -- | VAR | Correspondent to Pn083 | INT16 | RW | NO | ● | | | | | |
| 3054 | -- | VAR | Correspondent to Pn084 | INT16 | RW | NO | ● | | | | | |
| 3055 | -- | VAR | Correspondent to Pn085 | INT16 | RW | NO | ● | | | | | |
| 3056 | -- | VAR | Correspondent to Pn086 | INT16 | RW | NO | ● | | | | | |
| 3057 | -- | VAR | Correspondent to Pn087 | INT16 | RW | NO | ● | | | | | |
| 3058 | -- | VAR | Correspondent to Pn088 | INT16 | RW | NO | ● | | | | | |
| 3059 | -- | VAR | Correspondent to Pn089 | INT16 | RW | NO | ● | | | | | |
| 305A | -- | VAR | Correspondent to Pn090 | INT16 | RW | NO | ● | | | | | |
| 305B | -- | VAR | Correspondent to Pn091 | INT16 | RW | NO | ● | | | | | |
| 305C | -- | VAR | Correspondent to Pn092 | INT16 | RW | NO | ● | | | | | |
| 305D | -- | VAR | Correspondent to Pn093 | INT16 | RW | NO | ● | | | | | |
| 305E | -- | VAR | Correspondent to Pn094 | INT16 | RW | NO | ● | | | | | |
| 305F | -- | VAR | Correspondent to Pn095 | INT16 | RW | NO | ● | | | | | |
| 3060 | -- | VAR | Correspondent to Pn096 | UINT16 | RW | NO | ● | | | | | |
| 3061 | -- | VAR | Correspondent to Pn097 | UINT16 | RW | NO | ● | | | | | |
| 3062 | | VAR | Correspondent to Pn098 | UINT16 | RW | NO | ● | | | | | |
| 3063 | -- | VAR | Correspondent to Pn099 | UINT16 | RW | NO | ● | | | | | |
| 3064 | -- | VAR | Correspondent to Pn100 | UINT16 | RW | NO | ● | | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | Support | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 3065 | -- | VAR | Correspondent to Pn101 | UINT16 | RW | NO | ● | | | | | |
| 3066 | -- | VAR | Correspondent to Pn102 | UINT16 | RW | NO | ● | | | | | |
| 3067 | -- | VAR | Correspondent to Pn103 | UINT16 | RW | NO | ● | | | | | |
| 3068 | -- | VAR | Correspondent to Pn104 | UINT16 | RW | NO | ● | | | | | |
| 3069 | -- | VAR | Correspondent to Pn105 | UINT16 | RW | NO | ● | | | | | |
| 306A | -- | VAR | Correspondent to Pn106 | UINT16 | RW | NO | ● | | | | | |
| 306B | -- | VAR | Correspondent to Pn107 | UINT16 | RW | NO | ● | | | | | |
| 306C | -- | VAR | Correspondent to Pn108 | UINT16 | RW | NO | ● | | | | | |
| 306D | -- | VAR | Correspondent to Pn109 | UINT16 | RW | NO | ● | | | | | |
| 306E | -- | VAR | Correspondent to Pn110 | UINT16 | RW | NO | ● | | | | | |
| 306F | -- | VAR | Correspondent to Pn111 | UINT16 | RW | NO | ● | | | | | |
| 3070 | -- | VAR | Correspondent to Pn112 | UINT16 | RW | NO | ● | | | | | |
| 3071 | -- | VAR | Correspondent to Pn113 | UINT16 | RW | NO | ● | | | | | |
| 3072 | -- | VAR | Correspondent to Pn114 | UINT16 | RW | NO | ● | | | | | |
| 3073 | -- | VAR | Correspondent to Pn115 | UINT16 | RW | NO | ● | | | | | |
| 3074 | -- | VAR | Correspondent to Pn116 | UINT16 | RW | NO | ● | | | | | |
| 3075 | -- | VAR | Correspondent to Pn117 | UINT16 | RW | NO | ● | | | | | |
| 3076 | -- | VAR | Correspondent to Pn118 | UINT16 | RW | NO | ● | | | | | |
| 3077 | -- | VAR | Correspondent to Pn119 | UINT16 | RW | NO | ● | | | | | |
| 3078 | -- | VAR | Correspondent to Pn120 | UINT16 | RW | NO | ● | | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 6040 | -- | VAR | controlword | UINT16 | RW | YES | ● | | | | |
| 6041 | -- | VAR | statusword | UINT16 | RO | YES | ● | | | | |
| 605A | -- | VAR | quick_stop_option_code | INT16 | RW | NO | ● | | | | |
| 605B | -- | VAR | shutdown_option_code | INT16 | RW | NO | ● | | | | |
| 605C | -- | VAR | disable_operation_option_code | INT16 | RW | NO | ● | | | | |
| 605D | -- | VAR | stop_option_code | INT16 | RW | NO | ● | | | | |
| 605E | -- | VAR | fault_reaction_option_code | UINT16 | RW | NO | ● | | | | |
| 6060 | -- | VAR | modes_of_operation | INT8 | RW | YES | ● | | | | |
| 6061 | -- | VAR | modes_of_operation_display | INT8 | RO | YES | ● | | | | |
| 6062 | -- | VAR | position_demand_value | INT32 | RO | YES | | | ● | | |
| 6063 | -- | VAR | position_actual_value* | INT32 | RO | YES | | | ● | | |
| 6064 | -- | VAR | position_actual_value | INT32 | RO | YES | | | ● | | |
| 6065 | -- | VAR | following_error_window | UINT32 | RW | YES | | | ● | | |
| 6066 | -- | VAR | following_error_time_out | UINT16 | RW | YES | | | ● | | |
| 6067 | -- | VAR | position_window | UINT32 | RW | YES | | | ● | | |
| 6068 | -- | VAR | position_window_time | UINT16 | RW | YES | | | ● | | |
| 6069 | -- | VAR | velocity_sensor_actual_value | UINT16 | RO | YES | | | | ● | |
| 606B | -- | VAR | velocity_demand_value | INT32 | RO | YES | | | | ● | |
| 606C | -- | VAR | velocity_actual_value | INT32 | RO | YES | | | | ● | |
| 606D | -- | VAR | velocity_window | UINT16 | RW | YES | | | | ● | |
| 606E | -- | VAR | velocity_window_time | UINT16 | RW | YES | | | | ● | |
| 606F | -- | VAR | velocity_threshold | UINT16 | RW | YES | | | | ● | |
| 6070 | -- | VAR | velocity_threshold_time | UINT16 | RW | YES | | | | ● | |
| 607A | -- | VAR | target_position | INT32 | RW | YES | | | ● | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO mapping | Support | | | | unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | PP | PV | HM | |
| 607B | -- | ARRAY | position_range_limit | -- | -- | -- | | | ● | | |
| | 0 | | number_of_entries | UINT8 | RW | NO | | | ● | | |
| | 1 | | min_position_range_limit | INT32 | RW | NO | | | ● | | |
| | 2 | | max_position_range_limit | INT32 | RW | NO | | | ● | | |
| 607C | -- | VAR | home_offset | INT32 | RW | YES | | | ● | | ● |
| 607D | -- | ARRAY | Software_position _limit | -- | -- | -- | | | ● | | |
| | 0 | | number_of_entries | UINT8 | RW | NO | | | ● | | |
| | 1 | | min_position_limit | INT32 | RW | NO | | | ● | | |
| | 2 | | max_position _limit | INT32 | RW | NO | | | ● | | |
| 607E | -- | VAR | polarity | UINT8 | RW | NO | | | ● | ● | |
| 6081 | -- | VAR | profile_velocity | UINT32 | RW | YES | | | ● | | |
| 6083 | -- | VAR | profile_acceleration | UINT32 | RW | YES | | | ● | ● | |
| 6084 | -- | VAR | profile_deceleration | UINT32 | RW | YES | | | ● | ● | |
| 6085 | -- | VAR | quick_stop_deceleration | UINT32 | RW | YES | | | ● | ● | |
| 6086 | -- | VAR | motion_profile_type | INT16 | RW | YES | | | ● | ● | |
| 6093 | -- | ARRAY | position_factor | -- | -- | -- | | | ● | | ● |
| | 0 | | number_of_entries | UINT32 | RW | NO | | | ● | | ● |
| | 1 | | numerator | UINT32 | RW | NO | | | ● | | ● |
| | 2 | | divisor | UINT32 | RW | NO | | | ● | | ● |
| 6094 | -- | ARRAY | velocity_encoder_factor | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT32 | RW | NO | ● | | | | |
| | 1 | | numerator | UINT32 | RW | NO | ● | | | | |
| | 2 | | divisor | UINT32 | RW | NO | ● | | | | |
| 6097 | -- | ARRAY | acceleration_factor | -- | -- | -- | ● | | | | |
| | 0 | | number_of_entries | UINT32 | RW | NO | ● | | | | |
| | 1 | | numerator | UINT32 | RW | NO | ● | | | | |
| | 2 | | divisor | UINT32 | RW | NO | ● | | | | |

| Index | Subindex | Object | Name | Type | Attr. | PDO | Support | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | All | IP | PP | PV | HM | |
| 6098 | -- | VAR | homing_method | INT8 | RW | YES | | | | | ● | |
| 6099 | -- | ARRAY | homing_speeds | -- | -- | -- | | | | | ● | speed units |
| | 0 | | number_of_entries | UINT8 | RW | NO | | | | | ● | |
| | 1 | | speed_during_search_for_switch | UINT32 | RW | NO | | | | | ● | speed units |
| | 2 | | speed_during_search_for_zero | UINT32 | RW | NO | | | | | ● | speed units |
| 609A | -- | VAR | homing_acceleration | UINT32 | RW | NO | | | | | ● | acceleration units |
| 60C0 | -- | VAR | Interpolation sub mode select | INT16 | RW | NO | | ● | | | | |
| 60C1 | -- | ARRAY | Interpolation data record | INT32 | RW | -- | | ● | | | | |
| | 0 | | number_of_entries | UINT8 | -- | NO | | ● | | | | |
| | 1 | | the first parameter of ip function $f_{ip}(x1, .. xN)$ | see 60C0h | RW | YES | | ● | | | | position units |
| | 2 | | the second parameter of ip function $f_{ip}(x1, .. xN)$ | | RW | YES | | ● | | | | position units |
| 60C2 | -- | RECORD | Interpolation time period | UINT8 | -- | -- | | ● | | | | |
| | 0 | | number_of_entries | UINT8 | RO | NO | | ● | | | | |
| | 1 | | ip time units | UINT8 | RW | NO | | ● | | | | |
| | 2 | | ip time index | UINT8 | RW | NO | | ● | | | | |